VISIT…

Making Everything Easier!™

3rd Edition

Hacking

FOR DUMMIES®

Learn to:

- Use the latest ethical hacking methods and tools
- Test your Windows® or Linux® systems
- Hack databases, VoIP systems, and Web applications
- Report vulnerabilities and improve information security

Kevin Beaver, CISSP
Information Security Consultant

Get More and Do More at Dummies.com®

Start with **FREE** Cheat Sheets

Cheat Sheets include

- Checklists
- Charts
- Common Instructions
- And Other Good Stuff!

To access the Cheat Sheet created specifically for this book, go to
www.dummies.com/cheatsheet/hacking

Get Smart at Dummies.com

Dummies.com makes your life easier with 1,000s of answers on everything from removing wallpaper to using the latest version of Windows.

Check out our

- Videos
- Illustrated Articles
- Step-by-Step Instructions

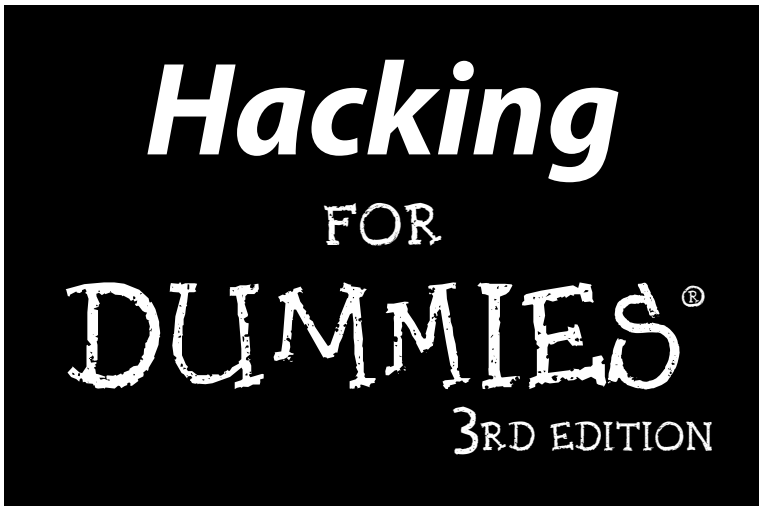
Plus, each month you can win valuable prizes by entering our Dummies.com sweepstakes. *

Want a weekly dose of Dummies? Sign up for Newsletters on

- Digital Photography
- Microsoft Windows & Office
- Personal Finance & Investing
- Health & Wellness
- Computing, iPods & Cell Phones
- eBay
- Internet
- Food, Home & Garden

Find out "HOW" at Dummies.com

**Sweepstakes not currently available in all countries; visit Dummies.com for official rules.*



by Kevin Beaver

Foreword by Stuart McClure

Wiley Publishing, Inc.

Hacking For Dummies®, 3rd Edition

Published by
Wiley Publishing, Inc.
111 River Street
Hoboken, NJ 07030-5774
www.wiley.com

Copyright © 2010 by Wiley Publishing, Inc., Indianapolis, Indiana

Published by Wiley Publishing, Inc., Indianapolis, Indiana

Published simultaneously in Canada

No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, except as permitted under Sections 107 or 108 of the 1976 United States Copyright Act, without either the prior written permission of the Publisher, or authorization through payment of the appropriate per-copy fee to the Copyright Clearance Center, 222 Rosewood Drive, Danvers, MA 01923, (978) 750-8400, fax (978) 646-8600. Requests to the Publisher for permission should be addressed to the Permissions Department, John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, (201) 748-6011, fax (201) 748-6008, or online at http://www.wiley.com/go/permissions.

Trademarks: Wiley, the Wiley Publishing logo, For Dummies, the Dummies Man logo, A Reference for the Rest of Us!, The Dummies Way, Dummies Daily, The Fun and Easy Way, Dummies.com, Making Everything Easier, and related trade dress are trademarks or registered trademarks of John Wiley & Sons, Inc. and/or its affiliates in the United States and other countries, and may not be used without written permission. All other trademarks are the property of their respective owners. Wiley Publishing, Inc., is not associated with any product or vendor mentioned in this book.

LIMIT OF LIABILITY/DISCLAIMER OF WARRANTY: THE PUBLISHER AND THE AUTHOR MAKE NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE ACCURACY OR COMPLETENESS OF THE CONTENTS OF THIS WORK AND SPECIFICALLY DISCLAIM ALL WARRANTIES, INCLUDING WITHOUT LIMITATION WARRANTIES OF FITNESS FOR A PARTICULAR PURPOSE. NO WARRANTY MAY BE CREATED OR EXTENDED BY SALES OR PROMOTIONAL MATERIALS. THE ADVICE AND STRATEGIES CONTAINED HEREIN MAY NOT BE SUITABLE FOR EVERY SITUATION. THIS WORK IS SOLD WITH THE UNDERSTANDING THAT THE PUBLISHER IS NOT ENGAGED IN RENDERING LEGAL, ACCOUNTING, OR OTHER PROFESSIONAL SERVICES. IF PROFESSIONAL ASSISTANCE IS REQUIRED, THE SERVICES OF A COMPETENT PROFESSIONAL PERSON SHOULD BE SOUGHT. NEITHER THE PUBLISHER NOR THE AUTHOR SHALL BE LIABLE FOR DAMAGES ARISING HEREFROM. THE FACT THAT AN ORGANIZATION OR WEBSITE IS REFERRED TO IN THIS WORK AS A CITATION AND/OR A POTENTIAL SOURCE OF FURTHER INFORMATION DOES NOT MEAN THAT THE AUTHOR OR THE PUBLISHER ENDORSES THE INFORMATION THE ORGANIZATION OR WEBSITE MAY PROVIDE OR RECOMMENDATIONS IT MAY MAKE. FURTHER, READERS SHOULD BE AWARE THAT INTERNET WEBSITES LISTED IN THIS WORK MAY HAVE CHANGED OR DISAPPEARED BETWEEN WHEN THIS WORK WAS WRITTEN AND WHEN IT IS READ. FULFILLMENT OF EACH COUPON OFFER IS THE SOLE RESPONSIBILITY OF THE OFFEROR.

For general information on our other products and services, please contact our Customer Care Department within the U.S. at 877-762-2974, outside the U.S. at 317-572-3993, or fax 317-572-4002.

For technical support, please visit www.wiley.com/techsupport.

Wiley also publishes its books in a variety of electronic formats. Some content that appears in print may not be available in electronic books.

Library of Congress Control Number: 2009942371

ISBN: 978-0-470-55093-9

Manufactured in the United States of America

10 9 8 7 6 5 4 3 2 1

About the Author

Kevin Beaver is an independent information security consultant, expert witness, keynote speaker, and author with Atlanta-based Principle Logic, LLC. He has over two decades of experience and specializes in performing information security assessments for Fortune 1000 corporations, security product vendors, independent software developers, universities, government agencies, nonprofit organizations, and small businesses. Before starting his information security consulting practice in 2001, Kevin served in various information technology and security roles for several healthcare, e-commerce, financial, and educational institutions.

Kevin has appeared on CNN television as an information security expert and has been quoted in *The Wall Street Journal, Fortune Small Business, Women's Health,* and *Inc.* magazine's technology site IncTechnology.com. Kevin's work has also been referenced by the PCI Council in their *Data Security Standard Wireless Guidelines.* Over the years, Kevin has been a top-rated keynote speaker and seminar leader and has presented at shows for IDC, RSA, CSI, IIA, ISSA, ISACA, and SecureWorld Expo more than 100 times. Additionally, he has performed over three dozen webcasts for TechTarget, Ziff-Davis, and other publishers.

Kevin has authored/co-authored seven information security books, including *Hacking Wireless Networks For Dummies*, *Securing the Mobile Enterprise For Dummies*, *Laptop Encryption For Dummies*, *The Definitive Guide to Email Management and Security* (RealtimePublishers.com), and *The Practical Guide to HIPAA Privacy and Security Compliance* (Auerbach). Kevin has written 18 whitepapers and more than 350 articles and is a regular contributor to SearchCompliance.com, SearchSoftwareQuality.com, SearchEnterpriseDesktop.com, SearchWindowsServer.com, SearchWinIT.com, and *Security Technology Executive* magazine. He has also written for CSOonline.com, Computerworld.com, and *Information Security* magazine.

Kevin is the creator and producer of the audio series *Security On Wheels* providing security learning for IT professionals on the go (SecurityOnWheels.com) and its associated blog (SecurityOnWheels.com/blog). He also rants about information security on Twitter at `www.twitter.com/kevinbeaver`. Kevin earned his bachelor's degree in Computer Engineering Technology from Southern College of Technology and his master's degree in Management of Technology from Georgia Tech. He has been a CISSP since 2001 and also holds MCSE, Master CNE, and IT Project+ certifications. Kevin can be reached through his Web sites at `www.principlelogic.com` and `http://securityonwheels.com`.

Dedication

Mom, this one's for you. You've been so strong fighting your cancer and have no idea how much of an inspiration you've been to me. I love you.

Author's Acknowledgments

First, I want to thank Amy, Garrett, and Mary Lin for being here for me and supporting me during the long hours I put into this edition. You all are the best! I'd like to thank Melody Layne, my original acquisitions editor at Wiley, for contacting me long ago with this book idea and providing me this great opportunity. I'd also like to thank my new acquisitions editor, Amy Fandrei, for continuing this project and presenting me the opportunity to shape this book into something I'm very proud of.

I'd like to thank my project editor, Jean Nelson. Yet again, you've been more than a pleasure to work with and have added a lot of value to this book. I'd also like to thank Brian Walls, my copy editor, for keeping my focus (and English) in line. Also, many thanks to my technical editor, business colleague, friend, and co-author of *Hacking Wireless Networks For Dummies,* Peter T. Davis. Again, I'm honored to be working with you and very much appreciate your valuable feedback. Your keen eye has really kept me in check.

Thanks to Ira Winkler and Jack Wiles for following up with me regarding my case study requests. Also, many thanks for Joshua Wright and Chip Andrews for contributing new case study material. You guys have really contributed some valuable content to this book.

Much gratitude to Joe Yeager formerly with HP's Application Security Center; Robert Abela with Acunetix; Chia-Chee Kuan with AirMagnet; Vladimir Katalov with Elcomsoft; Tony Haywood with Karalon; Victoria Muscat Inglott formerly with GFI Software; Kirk Thomas with Northwest Performance Software; David Vest with Mythicsoft; Thiago Zaninotti with N-Stalker; Mike Andrews and Chris Neppes with Port80 Software; Michael Berg with TamoSoft; Terry Ingoldsby with Amenaza Technologies; Amit Goyal and Fern Edison with Identity Finder for responding to all my requests. Much gratitude to all the others I forgot to mention as well!

Mega thanks to Queensrÿche, Rush, and Triumph for your energizing sounds and inspirational words. Yet again, your music helped me through the long days getting this new edition out. I wouldn't have wanted to do it without you! Thanks again to Neal Boortz for going against the grain and educating me about what's happening in our country and the world we live in. You keep me motivated as an entrepreneur, small business owner, and Libertarian. You speak the truth – keep it coming!

Thanks again to Brian Tracy for your immeasurable insight and guidance about what it takes to be a better person. Your contributions have helped me in so many ways — both personally and professionally.

Finally, I want to send out many thanks and humble appreciation to my clients for hiring me, a "no-name-brand" consultant, and keeping me around for the long term. I wouldn't be here without your willingness to break out of the "must hire big company" mindset and your continued support. Thank you very much.

Publisher's Acknowledgments

We're proud of this book; please send us your comments at http://dummies.custhelp.com. For other comments, please contact our Customer Care Department within the U.S. at 877-762-2974, outside the U.S. at 317-572-3993, or fax 317-572-4002.

Some of the people who helped bring this book to market include the following:

Acquisitions, Editorial

Project Editor: Jean Nelson

Acquisitions Editor: Amy Fandrei

Copy Editor: Brian Walls

Technical Editor: Peter T. Davis

Editorial Manager: Kevin Kirschner

Media Development Project Manager: Laura Moss-Hollister

Media Development Assistant Project Manager: Jenny Swisher

Media Development Associate Producers: Josh Frank, Marilyn Hummel, Douglas Kuhn, and Shawn Patrick

Editorial Assistant: Amanda Graham

Sr. Editorial Assistant: Cherie Case

Cartoons: Rich Tennant (www.the5thwave.com)

Composition Services

Project Coordinator: Sheree Montgomery

Layout and Graphics: Samantha K. Cherolis, Joyce Haughey, Ronald G. Terry

Proofreaders: Lindsay Littrell, Linda Seifert

Indexer: BIM Indexing & Proofreading Services

Special Help: Beth Stanton

Publishing and Editorial for Technology Dummies

Richard Swadley, Vice President and Executive Group Publisher

Andy Cummings, Vice President and Publisher

Mary Bednarek, Executive Acquisitions Director

Mary C. Corder, Editorial Director

Publishing for Consumer Dummies

Diane Graves Steele, Vice President and Publisher

Composition Services

Debbie Stailey, Director of Composition Services

Contents at a Glance

Foreword xix

Introduction 1

Part I: Building the Foundation for Ethical Hacking 7

Chapter 1: Introduction to Ethical Hacking 9
Chapter 2: Cracking the Hacker Mindset 25
Chapter 3: Developing Your Ethical Hacking Plan 35
Chapter 4: Hacking Methodology 45

Part II: Putting Ethical Hacking in Motion 59

Chapter 5: Social Engineering 61
Chapter 6: Physical Security 75
Chapter 7: Passwords 85

Part III: Hacking the Network 115

Chapter 8: Network Infrastructure 117
Chapter 9: Wireless LANs 151

Part IV: Hacking Operating Systems 179

Chapter 10: Windows 181
Chapter 11: Linux 207
Chapter 12: Novell NetWare 229

Part V: Hacking Applications 247

Chapter 13: Communication and Messaging Systems 249
Chapter 14: Web Sites and Applications 277
Chapter 15: Databases and Storage Systems 303

Part VI: Ethical Hacking Aftermath 315

Chapter 16: Reporting Your Results 317
Chapter 17: Plugging Security Holes 323
Chapter 18: Managing Security Changes 329

Part VII: The Part of Tens 335
Chapter 19: Ten Tips for Getting Upper Management Buy-In 337
Chapter 20: Ten Reasons Hacking Is the Only Effective Way to Test 343
Chapter 21: Ten Deadly Mistakes 347
Appendix: Tools and Resources 351
Index 367

Table of Contents

Foreword ..... *xix*

Introduction *1*

Who Should Read This Book? 1
About This Book 2
How to Use This Book 2
What You Don't Need to Read 3
Foolish Assumptions 3
How This Book Is Organized 3
Part I: Building the Foundation for Ethical Hacking 4
Part II: Putting Ethical Hacking in Motion 4
Part III: Hacking the Network 4
Part IV: Hacking Operating Systems 4
Part V: Hacking Applications 5
Part VI: Ethical Hacking Aftermath 5
Part VII: The Part of Tens 5
Icons Used in This Book 6
Where to Go from Here 6

Part I: Building the Foundation for Ethical Hacking *7*

Chapter 1: Introduction to Ethical Hacking 9

Straightening Out the Terminology 9
Defining hacker 10
Defining malicious user 11
Recognizing How Malicious Attackers Beget Ethical Hackers 11
Ethical hacking versus auditing 12
Policy considerations 12
Compliance and regulatory concerns 12
Understanding the Need to Hack Your Own Systems 13
Understanding the Dangers Your Systems Face 14
Nontechnical attacks 14
Network infrastructure attacks 15
Operating system attacks 15
Application and other specialized attacks 16
Obeying the Ethical Hacking Commandments 16
Working ethically 16
Respecting privacy 17
Not crashing your systems 17

Using the Ethical Hacking Process 17
Formulating your plan 18
Selecting tools 20
Executing the plan 22
Evaluating results 22
Moving on 23

Chapter 2: Cracking the Hacker Mindset 25

What You're Up Against 25
Who Breaks into Computer Systems 27
Why They Do It 29
Planning and Performing Attacks 32
Maintaining Anonymity 34

Chapter 3: Developing Your Ethical Hacking Plan 35

Establishing Your Goals 36
Determining Which Systems to Hack 37
Creating Testing Standards 40
Timing 40
Specific tests 41
Blind versus knowledge assessments 42
Location 43
Reacting to vulnerabilities you find 43
Silly assumptions 43
Selecting Security Assessment Tools 44

Chapter 4: Hacking Methodology 45

Setting the Stage for Testing 45
Seeing What Others See 47
Gathering public information 47
Mapping the network 50
Scanning Systems 52
Hosts 52
Open ports 53
Determining What's Running on Open Ports 53
Assessing Vulnerabilities 55
Penetrating the System 57

Part II: Putting Ethical Hacking in Motion 59

Chapter 5: Social Engineering 61

Social Engineering 101 61
Before You Start 62
Why Attackers Use Social Engineering 64

Understanding the Implications 65
Performing Social Engineering Attacks 66
Phishing for information 66
Building trust 68
Exploiting the relationship 69
Social Engineering Countermeasures 72
Policies 72
User awareness and training 73

Chapter 6: Physical Security 75

Physical Security Vulnerabilities 76
What to Look For 78
Building infrastructure 78
Utilities 79
Office layout and usage 80
Network components and computers 82

Chapter 7: Passwords 85

Password Vulnerabilities 86
Organizational password vulnerabilities 86
Technical password vulnerabilities 88
Cracking Passwords 89
Cracking passwords the old-fashioned way 89
High-tech password cracking 91
Password-protected files 102
Other ways to crack passwords 103
General Password-Cracking Countermeasures 109
Storing passwords 110
Policy considerations 110
Other considerations 111
Securing Operating Systems 113
Windows 113
Linux and UNIX 114

Part III: Hacking the Network 115

Chapter 8: Network Infrastructure 117

Network Infrastructure Vulnerabilities 119
Choosing Tools 120
Scanners and analyzers 120
Vulnerability assessment 121
Scanning, Poking, and Prodding 121
Port scanners 122
SNMP scanning 128
Banner grabbing 130

Firewall rules 131
Network analyzers 134
The MAC-daddy attack 140
Denial of service 145
Common Router, Switch, and Firewall Weaknesses 147
Unsecured interfaces 147
IKE weaknesses 148
General Network Defenses 149

Chapter 9: Wireless LANs 151

Understanding the Implications of Wireless Network Vulnerabilities 152
Choosing Your Tools 154
Wireless LAN Discovery 156
Checking for worldwide recognition 156
Scanning your local airwaves 157
Wireless Network Attacks and Countermeasures 158
Encrypted traffic 160
Countermeasures against encrypted traffic attacks 164
Rogue wireless devices 165
Countermeasures against rogue wireless devices 170
MAC spoofing 170
Countermeasures against MAC spoofing 175
Queensland DoS attack 175
Countermeasures against DoS attacks 176
Physical security problems 176
Countermeasures against physical security problems 176
Vulnerable wireless workstations 177
Countermeasures against vulnerable wireless workstations 177
Default configuration settings 178
Countermeasures against default configuration settings exploits 178

Part IV: Hacking Operating Systems 179

Chapter 10: Windows 181

Windows Vulnerabilities 182
Choosing Tools 183
Free Microsoft tools 183
All-in-one assessment tools 184
Task-specific tools 184

Information Gathering 185
System scanning 185
NetBIOS 187
Null Sessions 190
Mapping 191
Gleaning information 192
Countermeasures against null session hacks 194
Share Permissions 196
Windows defaults 196
Testing 197
Missing Patch Exploitation 198
Using Metasploit 200
Countermeasures against missing patch vulnerability exploits 205
Authenticated Scans 205

Chapter 11: Linux 207

Linux Vulnerabilities 208
Choosing Tools 208
Information Gathering 209
System scanning 209
Countermeasures against system scanning 213
Unneeded and Unsecured Services 213
Searches 213
Countermeasures against attacks on unneeded services 216
.rhosts and hosts.equiv Files 218
Hacks using the .rhosts and hosts.equiv files 218
Countermeasures against .rhosts and hosts.equiv file attacks 219
NFS 220
NFS hacks 220
Countermeasures against NFS attacks 221
File Permissions 221
File permission hacks 222
Countermeasures against file permission attacks 222
Buffer Overflows 223
Attacks 223
Countermeasures against buffer-overflow attacks 223
Physical Security 224
Physical security hacks 224
Countermeasures against physical security attacks 224
General Security Tests 225
Patching Linux 226
Distribution updates 227
Multiplatform update managers 227

Chapter 12: Novell NetWare 229

NetWare Vulnerabilities 229
Choosing Tools 230
Getting Started 230
Server access methods 231
Port scanning 231
Authentication 233
rconsole 233
Server-console access 236
Intruder detection 237
Testing for rogue NLMs 238
Countermeasures against rogue NLM attacks 241
Cleartext packets 242
Solid Practices for Minimizing NetWare Security Risks 243
Rename admin 243
Disable eDirectory browsing 244
Remove bindery contexts 245
Audit the system 246
TCP/IP parameters 246
Patch 246

Part V: Hacking Applications 247

Chapter 13: Communication and Messaging Systems 249

Messaging System Vulnerabilities 249
E-Mail Attacks 252
E-mail bombs 252
Banners 255
SMTP attacks 257
General best practices for minimizing e-mail security risks 266
Instant Messaging 267
IM vulnerabilities 267
Countermeasures against IM vulnerabilities 268
Voice over IP 270
VoIP vulnerabilities 270
Countermeasures against VoIP vulnerabilities 276

Chapter 14: Web Sites and Applications 277

Choosing Your Web Application Tools 278
Web Vulnerabilities 280
Directory traversal 280
Countermeasures against directory traversals 282
Input filtering attacks 283
Countermeasures against input attacks 291

Default script attacks 292
Countermeasures against default script attacks 294
Unsecured login mechanisms 294
Countermeasures against unsecured login systems 297
General security scans for Web application vulnerabilities 297
Best Practices for Minimizing Web Security Risks 298
Obscurity 299
Firewalls 299
Source code analysis 300

Chapter 15: Databases and Storage Systems 303

Databases 303
Choosing tools 303
Finding databases on the network 304
Cracking database passwords 306
Scanning databases for vulnerabilities 307
Best Practices for Minimizing Database Security Risks 308
Storage Systems 309
Choosing tools 309
Finding storage systems on the network 310
Rooting out sensitive text in network files 310
Best Practices for Minimizing Storage Security Risks 313

Part VI: Ethical Hacking Aftermath 315

Chapter 16: Reporting Your Results 317

Pulling the Results Together 317
Prioritizing Vulnerabilities 319
Reporting Methods 320

Chapter 17: Plugging Security Holes 323

Turning Your Reports into Action 323
Patching for Perfection 324
Patch management 325
Patch automation 325
Hardening Your Systems 326
Assessing Your Security Infrastructure 327

Chapter 18: Managing Security Changes 329

Automating the Ethical Hacking Process 329
Monitoring Malicious Use 330
Outsourcing Ethical Hacking 332
Instilling a Security-Aware Mindset 333
Keeping Up with Other Security Issues 334

Part VII: The Part of Tens 335

Chapter 19: Ten Tips for Getting Upper Management Buy-In 337

Cultivate an Ally and Sponsor 337
Don't Be a FUDdy Duddy 337
Demonstrate How the Organization Can't Afford to Be Hacked 338
Outline the General Benefits of Ethical Hacking 339
Show How Ethical Hacking Specifically Helps the Organization 339
Get Involved in the Business 339
Establish Your Credibility 340
Speak on Management's Level 340
Show Value in Your Efforts 340
Be Flexible and Adaptable 341

Chapter 20: Ten Reasons Hacking Is the Only Effective Way to Test 343

The Bad Guys Are Thinking Bad Thoughts, Using Good Tools, and Developing New Attack Methods 343
IT Governance and Compliance Is More Than High-Level Checklist Audits 343
Ethical Hacking Complements Audits and Security Evaluations 344
Someone's Going to Ask How Secure Your Systems Are 344
The Law of Averages Is Working Against Businesses 344
Ethical Hacking Creates a Better Understanding of What the Business Is Up Against 344
If a Breach Occurs, You Have Something to Fall Back On 345
Ethical Hacking Brings Out the Worst in Your Systems 345
Ethical Hacking Combines the Best of Penetration Testing and Vulnerability Testing 345
Ethical Hacking Can Uncover Operational Weaknesses That Might Go Overlooked For Years 345

Chapter 21: Ten Deadly Mistakes 347

Not Getting Prior Approval in Writing 347
Assuming That You Can Find All Vulnerabilities during Your Tests 347
Assuming That You Can Eliminate All Security Vulnerabilities 348
Performing Tests Only Once 348
Thinking That You Know It All 348
Running Your Tests without Looking at Things from a Hacker's Viewpoint 349
Not Testing the Right Systems 349
Not Using the Right Tools 349
Pounding Production Systems at the Wrong Time 349
Outsourcing Testing and Not Staying Involved 350

Appendix: Tools and Resources 351

Bluetooth 351
Certifications 352
Databases 352
Exploit Tools 352
General Research Tools 353
Hacker Stuff 354
Keyloggers 354
Laws and Regulations 354
Linux 355
Live Toolkits 355
Log Analysis 355
Messaging 355
Miscellaneous Tools 356
NetWare 356
Networks 356
Password Cracking 358
Patch Management 359
Security Education and Learning Resources 360
Security Methods and Models 360
Source Code Analysis 361
Storage 361
System Hardening 361
User Awareness and Training 362
Voice over IP 362
Vulnerability Databases 363
Web Applications 363
Windows 364
Wireless Networks 365

Index *367*

Foreword

Little more than a decade ago, IT security was barely a newborn in diapers. With only a handful of security professionals in 1994, few practiced security and even fewer truly understood it. Security technologies amounted to little more than anti-virus software and packet filtering routers at that time. And the concept of a "hacker" came primarily from the Hollywood movie *WarGames;* or more often it referred to someone with a low golf score. As a result, just like Rodney Dangerfield, it got "no respect," and no one took it seriously. IT professionals saw it largely as a nuisance, to be ignored — that is until they were impacted by it.

Today, the number of Certified Information Systems Security Professionals (CISSP) has topped 61,000 (`www.isc2.org`) worldwide, and there are more security companies dotting the landscape than anyone could possibly remember. Today security technologies encompass everything from authentication and authorization to firewalls and VPNs. There are so many ways to address the security problem that it can cause more than a slight migraine simply considering the alternatives. And the term *hacker* has become a permanent part of our everyday vernacular — as defined in nearly daily headlines. The world (and its criminals) has changed dramatically.

So what does all this mean for you, the home/end-user or IT/security professional that is thrust into this dangerous online world every time you hit the power button on your computer? The answer is *everything.* The digital landscape is peppered with land mines that can go off with the slightest touch or, better yet, without any provocation whatsoever. Consider some simple scenarios:

- Simply plugging into the Internet without a properly configured firewall can get you hacked before the pizza is delivered, within 30 minutes or less.
- Opening an e-mail attachment from a family member, friend, or work colleague can install a back door on your system, allowing a hacker free access to your computer.
- Downloading and executing a file via your Internet Messaging (IM) program can turn your pristine desktop into a Centers for Disease Control (CDC) hotzone, complete with the latest alphabet soup virus.
- Browsing to an innocent (and trusted) Web site can completely compromise your computer, allowing a hacker to read your sensitive files or, worse, delete them.

Trust me when we say the likelihood of becoming an Internet drive-by statistic on the information superhighway is painfully real.

I am often asked, “Is the fear, uncertainty, and doubt (FUD) centered on cyber-terrorism justified? Can cyber-terrorists really affect our computer systems and our public infrastructure as some have prognosticated like new-age Nostradamus soothsayers?” The answer I always give is, “Unequivocally, yes.” The possibility of a digital Pearl Harbor is closer than many think. Organized terrorist cells like Al Qaeda are raided almost weekly, and when computers are discovered, their drives are filled with cyber-hacking plans, U.S. infrastructure blueprints, and instructions on attacking U.S. computer and infrastructure targets.

Do you believe the energy commission’s report about the biggest power outage in U.S history? The one that on August 14, 2003, left one-fifth of the U.S. population without power (about 50 million people) for over 12 hours? Do you believe that it has to do with untrimmed trees and faulty control processes? If you believe in Occam’s Razor, then yes, the simplest explanation is usually the correct one, but remember this: The power outage hit just three days after the Microsoft Blaster worm, one of the most vicious computer worms ever unleashed on the Internet, first hit. Coincidence? Perhaps.

Some of you may be skeptical, saying, “Well, if the threat is so real, why hasn’t something bad happened yet?” I respond simply, “If I had come to you on September 10, 2001, and said that in the near future people would use commercial airplanes as bombs to kill over 3,000 people in the matter of 5 hours, would you believe me?” I understand your skepticism. And you should be skeptical. But we are asking for your trust, and your faith, before something bad happens. Trust that we know the truth, we know what is possible, and we know the mind of the enemy. I think we can all agree on at least one thing, we cannot allow them to succeed.

Every minute of every day there are governments, organized crime, and hacker groups turning the doorknobs on your house looking for an unlocked entry. They are rattling the windows and circling your domicile, looking for a weakness, a vulnerability, or a way into your house. Are you going to let them in? Are you going to sit idly by and watch as they ransack your belongings, make use of your facilities, and desecrate your sanctuary? Or are you going to empower yourself, educate yourself, and prevent them from winning? The actions you take today will ultimately answer that question.

Do not despair, all hope is not lost. Increasing security is more of a mindset than anything else. Security is akin to working out. If you don’t do it regularly, it won’t become a part of your lifestyle. And if it doesn’t become a part of your lifestyle, it will quickly become something you can forgo and avoid. In other words, you won’t be fit. Same thing applies for security. If you don’t realize that it is a process, not a goal, then you will never make it part of your everyday wellness routine; as a result, it quickly becomes something you forgo and avoid. And if you avoid it, you will eventually be bit by it.

The greatest gift you can give yourself is that of education. What you don't know may not kill you, but it may seriously impact you or someone you care about. Knowing what you don't know is the real trick. And filling in the gaps of knowledge is paramount to preventing a significant attack. *Hacking For Dummies* can fill in those gaps. Kevin has done a remarkable job in presenting material that is valuable and unique in that it covers hacking methodologies for Windows, Novell, and Linux, as well as such little-covered topics as physical security, social engineering, and malware. The varied coverage of security topics in this book is what helps you more completely understand the minds of hackers and how they work, and it will ultimately be the singular reason you may avoid an attack in the future. Read it carefully. Learn from it. And practice what it says in every area you can.

Make no mistake; the digital battlefield is very real. It has no beginning, it has no ending, it has no boundaries, and it has no rules. Read this book, learn from it, and defend yourself, or we may lose this digital war.

Stuart McClure is the founder and co-author of the highly-popular Hacking Exposed *book series (McGraw-Hill) and founder, President, and Chief Technology Officer of Foundstone, Inc., a division of McAfee. He can be reached at* `stu@foundstone.com`.

Introduction

Welcome to *Hacking For Dummies,* 3rd Edition. This book outlines — in plain English — computer hacker tricks and techniques that you can use to assess the security of your information systems, find the security vulnerabilities that matter, and fix the weaknesses before criminal hackers and malicious users take advantage of them. This hacking is the professional, aboveboard, and legal type of security testing — which I call *ethical hacking* throughout the book.

Computer and network security is a complex subject and an ever-moving target. You must stay on top of it to ensure that your information is protected from the bad guys. That's where the tools and techniques outlined in this book can help.

You can implement all the security technologies and other best practices possible, and your information systems might be secure — as far as you know. However, until you understand how malicious attackers think, apply that knowledge, and use the right tools to assess your systems from their point of view, you can't get a true sense of how secure your information really is.

Ethical hacking — which encompasses formal and methodical *penetration testing, white hat hacking,* and *vulnerability testing* — is necessary to find security flaws and to help validate that your information systems are truly secure on an ongoing basis. This book provides you with the knowledge to implement an ethical hacking program successfully along with countermeasures that you can put in place to keep external hackers and malicious users out of your business.

Who Should Read This Book?

Disclaimer: If you choose to use the information in this book to hack or break into computer systems maliciously and without authorization, you're on your own. Neither I (the author) nor anyone else associated with this book shall be liable or responsible for any unethical or criminal choices that you might make and execute using the methodologies and tools that I describe. This book is intended solely for IT and information security professionals to test information security — either on your own systems or on a client's systems — in an authorized fashion.

Okay, now that that's out of the way, it's time for the good stuff! This book is for you if you're a network administrator, information security manager, security consultant, security auditor, compliance manager, or interested in finding out more about legally and ethically testing computer systems and IT operations to make things more secure.

As the ethical hacker performing well-intended information security assessments, you can detect and point out security holes that might otherwise be overlooked. If you're performing these tests on your systems, the information you uncover in your tests can help you win over management and prove that information security really is a business issue to be taken seriously. Likewise, if you're performing these tests for your clients, you can help find security holes that can be plugged before the bad guys have a chance to exploit them.

The information in this book helps you stay on top of the security game and enjoy the fame and glory from helping your organization and clients prevent bad things from happening to their information.

About This Book

Hacking For Dummies, 3rd Edition, is a reference guide on hacking your systems to improve security. The ethical hacking techniques are based on written and unwritten rules of computer system penetration testing, vulnerability testing, and information security best practices. This book covers everything from establishing your hacking plan to testing your systems to plugging the holes and managing an ongoing ethical hacking program. Realistically, for many networks, operating systems, and applications, thousands of possible hacks exist. I cover the major ones on various platforms and systems. Whether you need to assess security vulnerabilities on a small home office network, a medium-sized corporate network, or across large enterprise systems, *Hacking For Dummies,* 3rd Edition, provides the information you need.

How to Use This Book

This book includes the following features:

- Various technical and nontechnical hack attacks and their detailed methodologies
- Information security testing case studies from well-known information security experts
- Specific countermeasures to protect against hack attacks

Before you start hacking your systems, familiarize yourself with the information in Part I so you're prepared for the tasks at hand. The adage "if you fail to plan, you plan to fail" rings true for the ethical hacking process. You must get permission and have a solid game plan in place if you're going to be successful.

This material is not intended to be used for unethical or illegal hacking purposes to propel you from script kiddie to mega hacker. Rather, it is designed to provide you with the knowledge you need to hack your own or your clients' systems — ethically and legally — to enhance the security of the information involved.

What You Don't Need to Read

Depending on your computer and network configurations, you may be able to skip chapters. For example, if you aren't running Linux or wireless networks, you can skip those chapters.

Foolish Assumptions

I make a few assumptions about you, the aspiring information security professional:

- You're familiar with basic computer-, network-, and information-security–related concepts and terms.
- You have a basic understanding of what hackers and malicious users do.
- You have access to a computer and a network on which to use these techniques.
- You have access to the Internet to obtain the various tools used in the ethical hacking process.
- You have permission to perform the hacking techniques described in this book.

How This Book Is Organized

This book is organized into seven modular parts, so you can jump around from one part to another as needed. Each chapter provides practical methodologies and practices you can use as part of your ethical hacking efforts, including checklists and references to specific tools you can use, as well as resources on the Internet.

Part I: Building the Foundation for Ethical Hacking

This part covers the fundamental aspects of ethical hacking. It starts with an overview of the value of ethical hacking and what you should and shouldn't do during the process. You get inside the malicious mindset and discover how to plan your ethical hacking efforts. This part covers the steps involved in the ethical hacking process, including how to choose the proper tools.

Part II: Putting Ethical Hacking in Motion

This part gets you rolling with the ethical hacking process. It covers several well-known and widely used hack attacks, including social engineering and cracking passwords, to get your feet wet. This part covers the human and physical elements of security, which tend to be the weakest links in any information security program. After you plunge into these topics, you'll know the tips and tricks required to perform common general hack attacks against your systems, as well as specific countermeasures to keep your information systems secure.

Part III: Hacking the Network

Starting with the larger network in mind, this part covers methods to test your systems for various well-known network infrastructure vulnerabilities. From weaknesses in the TCP/IP protocol suite to wireless network insecurities, you find out how networks are compromised by using specific methods of flawed network communications, along with various countermeasures that you can implement to avoid becoming a victim. This part also includes case studies on some of the network hack attacks that are presented.

Part IV: Hacking Operating Systems

Practically all operating systems have well-known vulnerabilities that hackers often exploit. This part jumps into hacking three widely used operating systems: Windows, Linux, and NetWare. The hacking methods include scanning your operating systems for vulnerabilities and enumerating the specific hosts to gain detailed information. This part also includes information on exploiting well-known vulnerabilities in these operating systems, taking over operating

systems remotely, and specific countermeasures that you can implement to make your operating systems more secure. This part also includes case studies on operating system hack attacks.

Part V: Hacking Applications

Application security is gaining more visibility in the information security arena these days. An increasing number of attacks are aimed directly at various applications, which are often able to bypass firewalls, intrusion detection systems, and antivirus software. This part discusses hacking specific applications and databases, including coverage of e-mail systems, instant messaging, Voice over Internet Protocol (VoIP), and storage systems, along with practical countermeasures that you can put in place to make your systems more secure.

One of the most common network attacks is against Web applications. Practically every firewall lets Web traffic into and out of the network, so most attacks are against the millions of Web applications available to almost anyone. This part also covers Web application hack attacks, countermeasures, and some application hacking case studies for real-world security testing scenarios.

Part VI: Ethical Hacking Aftermath

After you perform your ethical hack attacks, what do you do with the information you gather? Shelve it? Show it off? How do you move forward? This part answers these questions and more. From developing reports for upper management to remediating the security flaws that you discover to establishing procedures for your ongoing ethical hacking efforts, this part brings the ethical hacking process full circle. This information not only ensures that your effort and time are well spent, but also is evidence that information security is an essential element for success in any business that depends on computers and information technology.

Part VII: The Part of Tens

This part contains tips to help ensure the success of your ethical hacking program. You find out how to get upper management to buy into your ethical hacking program so you can get going and start protecting your systems. This part also includes the top ten ethical hacking mistakes you absolutely must avoid.

This part also includes an appendix that provides a one-stop reference listing of ethical hacking tools and resources. You can find all the links in the Appendix on the *Hacking For Dummies* online Cheat Sheet at `www.dummies.com/cheatsheet/hacking`.

Icons Used in This Book

This icon points out technical information that is interesting but not vital to your understanding of the topic being discussed.

This icon points out information that is worth committing to memory.

This icon points out information that could have a negative impact on your ethical hacking efforts — so please read it!

This icon refers to advice that can help highlight or clarify an important point.

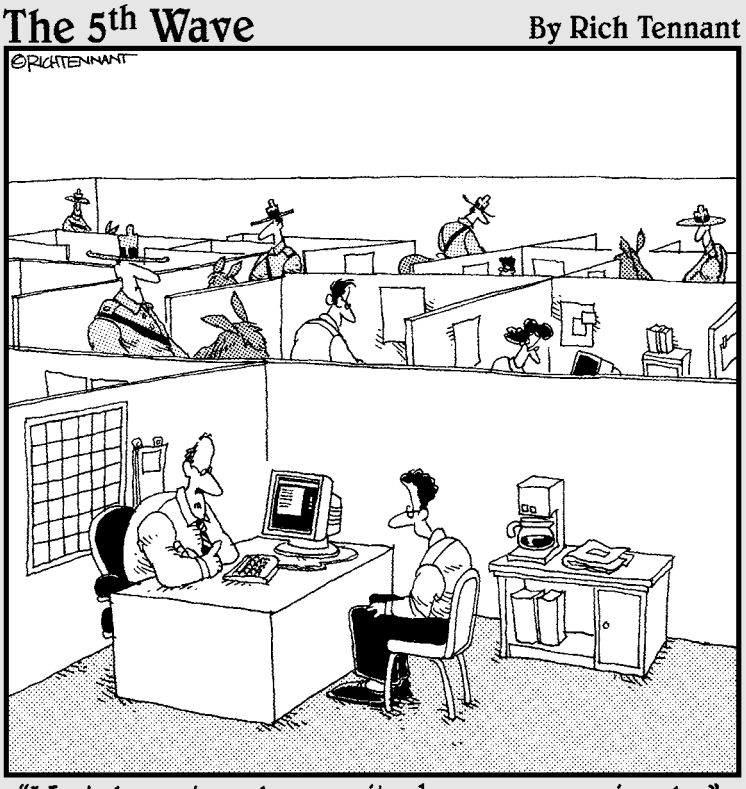
Where to Go from Here

The more you know about how external hackers and rogue insiders work and how your systems should be tested, the better you're able to secure your computer systems. This book provides the foundation that you need to develop and maintain a successful ethical hacking program for your organization and customers.

Keep in mind that the high-level concepts of ethical hacking won't change as often as the specific information security vulnerabilities you protect against. Ethical hacking will always remain an art and a science in a field that's ever-changing. You must keep up with the latest hardware and software technologies, along with the various vulnerabilities that come about month after month and year after year. You won't find a single *best* way to hack your systems, so tweak this information to your heart's content. Happy (ethical) hacking!

Part I

Building the Foundation for Ethical Hacking



"We take network security here very seriously."

In this part . . .

Your mission — should you choose to accept it — is to find the holes in your network before the bad guys do. This mission will be fun, educational, and most likely entertaining. It will certainly be an eye-opening experience. The cool part is that you can emerge as the hero, knowing that your organization will be better protected against malicious hacker and insider attacks and less likely to have its name smeared across the headlines.

If you're new to ethical hacking, this is the place to begin. The chapters in this part get you started with information on what to do and how to do it when you're hacking your own systems. Oh, and you find out what not to do. This information will guide you through building the foundation for your ethical hacking program to make sure you go down the right path and don't veer off down a one-way dead-end street. This mission is indeed possible — you just have to get your ducks in a row first.

Chapter 1

Introduction to Ethical Hacking

In This Chapter

- Understanding hackers' and malicious users' objectives
- Differentiating between ethical hackers and malicious attackers
- Examining how the ethical hacking process came about
- Understanding the dangers that your computer systems face
- Starting to use the ethical hacking process

This book is about hacking ethically — the methodology of testing your computers and networks for security vulnerabilities and plugging the holes you find before the bad guys get a chance to exploit them.

Although *ethical* is an often overused and misunderstood word, *Webster's New World Dictionary* defines *ethical* perfectly for the context of this book and the professional security testing techniques that I cover — that is, "conforming to the standards of conduct of a given profession or group." IT and information security practitioners are obligated to perform the tests covered in this book aboveboard and only after permission has been obtained by the owner(s) of the systems — hence the disclaimer in this book's Introduction.

Straightening Out the Terminology

Most people have heard of hackers and malicious users. Many have even suffered the consequences of hackers' criminal actions. So who are these people? And why do you need to know about them? The next few sections give you the lowdown on these attackers.

In this book, I use the following terminology:

- *Hackers* (or external attackers) try to compromise computers and sensitive information for ill-gotten gains — usually from the outside — as an unauthorized user. Hackers go for almost any system they think they can compromise. Some prefer prestigious, well-protected systems, but hacking into anyone's system increases an attacker's status in hacker circles.
- *Malicious internal users* (or internal attackers) try to compromise computers and sensitive information from the inside as authorized and "trusted" users. Malicious users go for systems they believe they can compromise for ill-gotten gains or revenge.

  Malicious attackers are, generally speaking, both hackers and malicious users. For the sake of simplicity, I refer to both as *hackers* and specify *hacker* or *malicious user* only when I need to drill down further into their tools, techniques, and ways of thinking.
- *Ethical hackers* (or good guys) hack systems to discover vulnerabilities to protect against unauthorized access, abuse, and misuse.

Defining hacker

Hacker has two meanings:

- Traditionally, hackers like to tinker with software or electronic systems. Hackers enjoy exploring and learning how computer systems operate. They love discovering new ways to work — both mechanically and electronically.
- In recent years, hacker has taken on a new meaning — someone who maliciously breaks into systems for personal gain. Technically, these criminals are *crackers* (criminal hackers). Crackers break into, or crack, systems with malicious intent. They are out for personal gain: fame, profit, and even revenge. They modify, delete, and steal critical information, often making other people miserable.

The good-guy *(white hat)* hackers don't like being in the same category as the bad-guy *(black hat)* hackers. (In case you're curious, the white hat and black hat terms come from old Western TV shows in which the good guys wore white cowboy hats and the bad guys wore black cowboy hats.) *Gray hat* hackers are a little bit of both. Whatever the case, most people have a negative connotation for the word *hacker*.

Many malicious hackers claim that they don't cause damage but instead help others. Yeah, right. Malicious hackers are electronic thieves and deserve the consequences of their actions.

Defining malicious user

Malicious users — meaning a rogue employee, contractor, intern, or other user who abuses his or her privileges — is a common term in security circles and in headlines about information breaches. A long-standing statistic states that insiders carry out 80% of all security breaches. Whether this number is accurate is still questionable, but based on what I've seen and numerous annual surveys, undoubtedly an insider problem makes up the majority of all computer breaches.

The issue is not necessarily users "hacking" internal systems, but rather users who abuse the computer access privileges they've been given. Users ferret through critical database systems to glean sensitive information, e-mail confidential client information to the competition or other third parties, or delete sensitive files from servers that they probably didn't need to have access to in the first place. There's also the occasional ignorant insider whose intent is not malicious but who still causes security problems by moving, deleting, or corrupting sensitive information.

Malicious users are often ethical hackers' worst enemies because they know exactly where to go to get the goods and don't need to be computer savvy to compromise sensitive information. These users have the access they need and the management trusts them without question.

Recognizing How Malicious Attackers Beget Ethical Hackers

You need protection from hacker shenanigans; you need (or need to become) an ethical hacker. An ethical hacker possesses the skills, mindset, and tools of a hacker but is also trustworthy. Ethical hackers perform the hacks as security tests for their systems based on how hackers might work.

Ethical hacking — which encompasses formal and methodical penetration testing, white hat hacking, and vulnerability testing — involves the same tools, tricks, and techniques that hackers use, but with one major difference: Ethical hacking is performed with the target's permission. The intent of ethical hacking is to discover vulnerabilities from a malicious attacker's viewpoint to better secure systems. Ethical hacking is part of an overall information risk management program that allows for ongoing security improvements. Ethical hacking can also ensure that vendors' claims about the security of their products are legitimate.

If you perform ethical hacking tests for clients or simply want to add another certification to your credentials, you might want to consider becoming a Certified Ethical Hacker (C|EH), through a certification program sponsored by EC-Council. See `www.eccouncil.org/CEH.htm` for more information.

Ethical hacking versus auditing

Many people confuse ethical hacking with security auditing but there are big differences. Security auditing involves comparing a company's security policies to what's actually taking place. The intent of security auditing is to validate that security controls exist — typically using a risk-based approach. Auditing often involves reviewing business processes and might not be very technical. I often refer to security audits as "security checklists" because they're usually based off (you guessed it) checklists.

Conversely, ethical hacking focuses on vulnerabilities that can be exploited. It validates that security controls *do not* exist. Ethical hacking can be both highly technical and nontechnical and, although you do use a formal methodology, it tends to be a bit less structured than formal auditing. If auditing continues to take place in your organization, you might consider integrating the ethical hacking techniques I outline into your auditing process.

Policy considerations

If you choose to make ethical hacking an important part of your business's risk management program, you really need to have a documented security testing policy. Such a policy outlines the type of ethical hacking that is done, which systems (such as servers, Web applications, laptops, and so on) are tested, and how often the testing is performed. Specific procedures for carrying out your security tests could outline the ethical hacking methodology I cover in this book. You might also consider creating a security standards document that outlines the specific security testing tools that are used and specific dates your systems are tested each year. You might list standard testing dates, such as once per quarter for external systems and biannual tests for internal systems.

Compliance and regulatory concerns

Your own internal policies might dictate how company management views security testing, but you also need to consider the state, federal, and global laws and regulations that affect your business. Many of the federal laws and regulations, such as the Health Insurance Portability and Accountability Act

(HIPAA), Gramm-Leach-Bliley Act (GLBA), North American Electric Reliability Corporation (NERC) CIP requirements, and Payment Card Industry Data Security Standard (PCI DSS) require periodic and consistent security evaluations. Incorporating your ethical hacking into these required tests is a great way to meet the state and federal regulations and beef up your overall privacy and security compliance program.

Understanding the Need to Hack Your Own Systems

To catch a thief, you must think like a thief. That's the basis for ethical hacking. Knowing your enemy is absolutely critical. See Chapter 2 for details about how malicious attackers work.

The law of averages works against security. With the increased number of hackers and their expanding knowledge, and the growing number of system vulnerabilities and other unknowns, eventually, all computer systems and applications will be hacked or compromised in some way. Protecting your systems from the bad guys — and not just the generic vulnerabilities that everyone knows about — is absolutely critical. When you know hacker tricks, you find out how vulnerable your systems really are.

Hacking preys on weak security practices and undisclosed vulnerabilities. Firewalls, encryption, and passwords can create a false feeling of safety. These security systems often focus on high-level vulnerabilities, such as basic access control, without affecting how the bad guys work. Attacking your own systems to discover vulnerabilities helps make them more secure. Ethical hacking is the only proven method of greatly hardening your systems from attack. If you don't identify weaknesses, it's only a matter of time before the vulnerabilities are exploited.

As hackers expand their knowledge, so should you. You must think like them and work like them to protect your systems from them. As the ethical hacker, you must know the activities that hackers carry out and how to stop their efforts. Knowing what to look for and how to use that information helps you to thwart hackers' efforts.

You don't have to protect your systems from *everything*. You can't. The only protection against everything is to unplug your computer systems and lock them away so no one can touch them — not even you. But doing so is not the best approach to information security and it's certainly not good for business. What's important is to protect your systems from known vulnerabilities and common attacks.

Anticipating all the possible vulnerabilities you'll have in your systems and business processes is impossible. You certainly can't plan for all possible attacks — especially the unknown ones. However, the more combinations you try and the more you test whole systems instead of individual units, the better your chances are of discovering vulnerabilities that affect your information systems in their entirety.

Don't take ethical hacking too far, though; hardening your systems from unlikely attacks makes little sense. For instance, if you don't have a lot of foot traffic in your office and no internal Web server running, you might not have as much to worry about as an Internet hosting provider might have. Your overall goals as an ethical hacker are

- Prioritize your systems so you can focus your efforts on what matters.
- Hack your systems in a nondestructive fashion.
- Enumerate vulnerabilities and, if necessary, prove to management that vulnerabilities exist and can be exploited.
- Apply results to remove the vulnerabilities and better secure your systems.

Understanding the Dangers Your Systems Face

It's one thing to know generally that your systems are under fire from hackers around the world and malicious users around the office; it's another to understand the specific attacks against your systems that are possible. This section offers some well-known attacks but is by no means a comprehensive listing.

Many information security vulnerabilities aren't critical by themselves. However, exploiting several vulnerabilities at the same time can take its toll on a system. For example, a default Windows OS configuration, a weak SQL Server administrator password, or a server hosted on a wireless network might not be major security concerns separately — but a hacker exploiting all three of these vulnerabilities at the same time could lead to sensitive information disclosure and more.

Nontechnical attacks

Exploits that involve manipulating people — end users and even yourself — are the greatest vulnerability within any computer or network infrastructure. Humans are trusting by nature, which can lead to social engineering exploits.

Social engineering is the exploitation of the trusting nature of human beings to gain information for malicious purposes. Check out Chapter 5 for more information about social engineering and how to guard your systems against it.

Other common and effective attacks against information systems are physical. Hackers break into buildings, computer rooms, or other areas containing critical information or property to steal computers, servers, and other valuable equipment. Physical attacks can also include *dumpster diving* — rummaging through trash cans and dumpsters for intellectual property, passwords, network diagrams, and other information.

Network infrastructure attacks

Hacker attacks against network infrastructures can be easy because many networks can be reached from anywhere in the world via the Internet. Some examples of network infrastructure attacks include the following:

- Connecting to a network through an unsecured wireless router attached behind a firewall
- Exploiting weaknesses in network protocols, such as TCP/IP and NetBIOS
- Flooding a network with too many requests, creating a denial of service (DoS) for legitimate requests
- Installing a network analyzer on a network and capturing every packet that travels across it, revealing confidential information in clear text

Operating system attacks

Hacking an operating system (OS) is a preferred method of the bad guys. OS attacks make up a large portion of hacker attacks simply because every computer has an operating system and OSes are susceptible to many well-known exploits.

Occasionally, some operating systems that tend to be more secure out of the box — such as Novell NetWare and OpenBSD— are attacked, and vulnerabilities turn up. But hackers often prefer attacking Windows and Linux because they are widely used and better known for their weaknesses.

Here are some examples of attacks on operating systems:

- Exploiting specific network protocol implementations
- Attacking built-in authentication systems
- Breaking file system security
- Cracking passwords and weak encryption implementations

Application and other specialized attacks

Applications take a lot of hits by hackers. Programs, such as e-mail server software and Web applications, are often beaten down:

- Hypertext Transfer Protocol (HTTP) and Simple Mail Transfer Protocol (SMTP) applications are frequently attacked because most firewalls and other security mechanisms are configured to allow full access to these services from the Internet.
- Voice over Internet Protocol (VoIP) faces increasing attacks as it finds its way into more and more businesses.
- Unsecured files containing sensitive information are scattered throughout workstation and server shares, and database systems contain numerous vulnerabilities that malicious users can exploit.

Ethical hackers carry out such attacks against computer systems, physical controls, and people and highlight any associated weaknesses. Parts II through V of this book cover these attacks in detail, along with specific countermeasures you can implement against attacks against your business.

Obeying the Ethical Hacking Commandments

Every ethical hacker must abide by a few basic commandments. If not, bad things can happen. I've seen these commandments ignored or forgotten when planning or executing ethical hacking tests. The results weren't positive — trust me.

Working ethically

The word ethical in this context means working with high professional morals and principles. Whether you're performing ethical hacking tests against your own systems or for someone who has hired you, everything you do as an ethical hacker must be aboveboard and must support the company's goals. No hidden agendas allowed!

Trustworthiness is the ultimate tenet. The misuse of information is absolutely forbidden. That's what the bad guys do. Let them receive a fine or go to prison because of their poor choices.

Respecting privacy

Treat the information you gather with the utmost respect. All information you obtain during your testing — from Web application log files to clear text passwords to personally identifiable information and beyond — must be kept private. Don't snoop into confidential corporate information or employees' private lives. If you sense that a colleague or team member breaches privacy and you feel like someone should know about it, consider sharing that information with the appropriate manager or project sponsor.

Involve others in your process. Employ a watch-the-watcher system that can help build trust and support for your ethical hacking projects.

Not crashing your systems

One of the biggest mistakes I've seen people make when trying to hack their own systems is inadvertently crashing the systems they're trying to keep running. Poor planning is the main cause of this mistake. These testers either have not read the documentation or misunderstand the usage and power of the security tools and techniques at their disposal.

Although it's not likely, you can create DoS conditions on your systems when testing. Running too many tests too quickly can cause system lockups, data corruption, reboots, and more. I should know: I've done it! Don't rush and assume that a network or specific host can handle the beating that network tools and vulnerability scanners can dish out.

Many vulnerability scanners can control how many tests are performed on a system at the same time. These tools are especially handy when you need to run the tests on production systems during regular business hours.

You can even accidentally create an account or system lockout condition by socially engineering someone into changing a password, not realizing the consequences of your actions.

Using the Ethical Hacking Process

Like practically any IT or security project, ethical hacking needs to be planned. It's been said that action without planning is at the root of every failure. Strategic and tactical issues in the ethical hacking process need to be determined and agreed upon. To ensure the success of your efforts, spend time up front planning for any amount of testing — from a simple password-cracking test to an all-out penetration test on a Web application.

If you choose to hire a "reformed" hacker to work with you during your testing or to obtain an independent perspective, be careful. I cover the pros, cons, do's, and don'ts associated with hiring an ethical hacker in Chapter 18.

Formulating your plan

Getting approval for ethical hacking is essential. Make sure that what you're doing is known and visible — at least to the decision makers. Obtaining *sponsorship* of the project is the first step. Sponsorship could come from your manager, an executive, your client, or even yourself if you're the boss. You need someone to back you up and sign off on your plan. Otherwise, your testing might be called off unexpectedly if someone claims you were never authorized to perform the tests.

The authorization can be as simple as an internal memo or an e-mail from your boss when you perform these tests on your own systems. If you're testing for a client, have a signed contract stating the client's support and authorization. Get written approval on this sponsorship as soon as possible to ensure that none of your time or effort is wasted. This documentation is your Get Out of Jail Free card if anyone questions what you're doing, or worse, if the authorities come calling. Don't laugh — it wouldn't be the first time it happened.

One slip can crash your systems — not necessarily what anyone wants. You need a detailed plan, but that doesn't mean you need volumes of testing procedures to make things overly complex. A well-defined scope includes the following information:

- **Specific systems to be tested:** When selecting systems to test, start with the most critical systems and processes or the ones you suspect are the most vulnerable. For instance, you can test server OS passwords, an Internet-facing Web application, or attempt social engineering attacks before drilling down into all your systems.
- **Risks involved:** Have a contingency plan for your ethical hacking process in case something goes awry. What if you're assessing your firewall or Web application and you take it down? This can cause system unavailability, which can reduce system performance or employee productivity. Even worse, it might cause loss of data integrity, loss of data itself, and even bad publicity. It'll most certainly tick off a person or two and make you look bad.

Handle social engineering and DoS attacks carefully. Determine how they affect the systems you test and your entire organization.

- **Dates the tests will be performed and your overall timeline:** Determining when the tests are performed is something that you must think long and hard about. Do you perform tests during normal business hours? How about late at night or early in the morning so that production systems aren't affected? Involve others to make sure they approve of your timing.

  The best approach is an *unlimited attack,* where any type of test is possible at any time of day. The bad guys aren't breaking into your systems within a limited scope, so why should you? Some exceptions to this approach are performing DoS attacks, social engineering, and physical security tests.

- **Knowledge of the systems you have before you start testing:** You don't need extensive knowledge of the systems you're testing — just a basic understanding. This basic understanding helps protect you and the tested systems.

  Understanding the systems you're testing shouldn't be difficult if you're hacking your own in-house systems. If you're testing a client's systems, you may have to dig deeper. In fact, I've only had one or two clients ask for a fully blind assessment. Most IT managers and others responsible for security are scared of these assessments — and they can take more time and cost more. Base the type of test you perform on your organization or client's needs.

- **Actions you will take when a major vulnerability is discovered:** Don't stop after you find one security hole. Keep going to see what else you can discover. I'm not saying to keep hacking until the end of time or until you crash all your systems; simply pursue the path you're going down until you can't hack it any longer (pun intended). If you haven't found any vulnerability, you haven't looked hard enough. If you uncover something big, you do need to share that information with the key players as soon as possible to plug the hole before it's exploited.

- **The specific deliverables:** This includes vulnerability scanner reports and a higher-level report outlining the important vulnerabilities to address, along with countermeasures to implement.

One of your goals might be to perform the tests without being detected. For example, you might perform your tests on remote systems or on a remote office, and you don't want the users aware of what you're doing. Otherwise, the users might catch on to you and be on their best behavior — instead of their normal behavior.

Selecting tools

As with any project, if you don't have the right tools for ethical hacking, you might have difficulty accomplishing the task effectively. Having said that, just because you use the right tools doesn't mean that you'll discover all the right vulnerabilities.

Know the personal and technical limitations. Many vulnerability scanners generate false positives and negatives (incorrectly identifying vulnerabilities). Others just skip right over vulnerabilities altogether. In certain situations, you might need to run multiple vulnerability scanners to find the most vulnerabilities.

Many tools focus on specific tests, and no tool can test for everything. For the same reason you wouldn't drive a nail with a screwdriver, don't use a word processor to scan your network for open ports. This is why you need a set of specific tools for the task. The more (and better) tools you have, the easier your ethical hacking efforts are.

Make sure you're using the right tool for the task:

- To crack passwords, you need cracking tools, such as ophcrack and Proactive Password Auditor.

  A general port scanner, such as SuperScan or Nmap, won't work for cracking passwords and rooting out detailed vulnerabilities.
- For an in-depth analysis of a Web application, a Web application assessment tool (such as N-Stalker or WebInspect) is more appropriate than a network analyzer (such as Wireshark).

When selecting the right security tool for the task, ask around. Get advice from your colleagues and from other people online. A simple groups search on Google (`http://groups.google.com`), LinkedIn (`www.linkedin.com`) or a perusal of security portals, such as `http://SecurityFocus.com` and `http://SearchSecurity.com`, often produces great feedback from other security experts about what works and what doesn't.

Hundreds, if not thousands, of tools can be used for ethical hacking — from software-based vulnerability scanner programs to hardware-based network analyzers. The following list runs down some of my favorite commercial, freeware, and open source security tools:

- Cain & Abel
- OmniPeek

- SuperScan
- QualysGuard
- WebInspect
- Proactive Password Auditor
- Metasploit
- LANguard
- AirMagnet WiFi Analyzer

I discuss these tools and many others in Parts II through V when I go into the specific hack attacks. Appendix A contains a more comprehensive listing of these tools for your reference.

The capabilities of many security and hacking tools are often misunderstood. This misunderstanding has cast a negative light on otherwise excellent and legitimate tools.

Some of these security testing tools are complex. Whichever tools you use, familiarize yourself with them before you start using them. Here are ways to do that:

- Read the readme and/or online help files and FAQs.
- Study the user's guides.
- Use the tools in a lab or test environment.
- Consider formal classroom training from the security tool vendor or another third-party training provider, if available.

Look for these characteristics in tools for ethical hacking:

- Adequate documentation
- Detailed reports on the discovered vulnerabilities, including how they might be exploited and fixed
- General industry acceptance
- Availability of updates and support
- High-level reports that can be presented to managers or nontechnical types

These features can save you a ton of time and effort when you're performing your tests and writing your final reports.

Executing the plan

Good ethical hacking takes persistence. Time and patience are important. Be careful when you're performing your ethical hacking tests. A hacker in your network or a seemingly benign employee looking over your shoulder might watch what's going on and use this information against you or your business.

Making sure that no hackers are on your systems before you start isn't practical. Be sure you keep everything as quiet and private as possible. This is especially critical when transmitting and storing your test results. If possible, encrypt any e-mails and files containing sensitive test information by using Pretty Good Privacy (PGP) (`www.pgp.com`), encrypted Zip file, or similar technology.

You're now on a reconnaissance mission. Harness as much information as possible about your organization and systems, much like malicious hackers do. Start with a broad view and narrow your focus:

1. Search the Internet for your organization's name, your computer and network system names, and your IP addresses.

   Google is a great place to start.

2. Narrow your scope, targeting the specific systems you're testing.

   Whether you're assessing physical security structures or Web applications, a casual assessment can turn up a lot of information about your systems.

3. Further narrow your focus with a more critical eye. Perform actual scans and other detailed tests to uncover vulnerabilities on your systems.

4. Perform the attacks and exploit any vulnerabilities you find, if that's what you choose to do.

Check out Chapter 4 to find out more information and tips on using this process.

Evaluating results

Assess your results to see what you've uncovered, assuming that the vulnerabilities haven't been made obvious before now. This is where knowledge counts. Your skill at evaluating the results and correlating the specific vulnerabilities discovered will get better with practice. You'll end up knowing your systems much better than anyone else. This makes the evaluation process much simpler moving forward.

Submit a formal report to upper management or to your client, outlining your results and any recommendations you wish to share. Keep these parties in the loop to show that your efforts and their money are well spent. Chapter 16 describes the ethical hacking reporting process.

Moving on

When you finish your ethical hacking tests, you (or your client) still need to implement your recommendations to make sure the systems are secure. Otherwise, all the time, money, and effort spent on ethical hacking goes to waste.

New security vulnerabilities continually appear. Information systems constantly change and become more complex. New hacker exploits and security vulnerabilities are regularly uncovered. You might even discover new ones yourself! Vulnerability scanners get better and better. Security tests are a snapshot of the security posture of your systems. At any time, everything can change, especially after upgrading software, adding computer systems, or applying patches. Plan to test regularly and consistently (for example, once a month, once a quarter, or biannually). Chapter 18 covers managing security changes.

Chapter 2

Cracking the Hacker Mindset

In This Chapter

- Understanding the enemy
- Profiling hackers and malicious users
- Understanding why attackers do what they do
- Examining how attackers go about their business

Before you start assessing the security of your own systems, you may want to know something about the people you're up against. Many information security product vendors and other professionals claim that you should protect your systems from the bad guys — both internal and external. But what does this mean? How do you know how these people think and work?

Knowing what hackers and malicious users want helps you understand how they work. Understanding how they work helps you to look at your information systems in a whole new way. In this chapter, I describe the challenges you face from hackers, the people actually doing the misdeeds, and their motivations and methods so you're better prepared for your ethical hacking tests.

What You're Up Against

Thanks to sensationalism in the media, public perception of *hacker* has transformed from harmless tinkerer to malicious criminal. Nevertheless, hackers often state that the public misunderstands them, which is mostly true. It's easy to prejudge what you don't understand. Unfortunately, many hacker stereotypes are based on misunderstanding rather than fact, misunderstanding that fuels a constant debate.

Hackers can be classified by both their abilities and their underlying motivations. Some are skilled, and their motivations are benign; they're merely seeking more knowledge. At the other end of the spectrum, hackers with malicious intent seek some form of personal gain. Unfortunately, the negative aspects of hacking usually overshadow the positive aspects and promote the negative stereotypes.

Historically, hackers hacked for the pursuit of knowledge and the thrill of the challenge. *Script kiddies* (hacker wannabes with limited skills) aside, hackers are adventurous and innovative thinkers and are always devising new ways to exploit computer vulnerabilities. (For more on script kiddies, see the section, "Who Breaks into Computer Systems," later in this chapter.) They see what others often overlook. They wonder what would happen if a cable was unplugged, a switch was flipped, or lines of code were changed in a program. These old-school hackers are like Tim "The Toolman" Taylor — Tim Allen's character on the classic sitcom *Home Improvement* — thinking they can improve electronic and mechanical devices by "rewiring them." More recent evidence shows that many hackers may also hack for political, competitive, and even financial purposes, so times are changing.

When they were growing up, hackers' rivals were monsters and villains on video game screens. Now hackers see their electronic foes as only that — electronic. Hackers who perform malicious acts don't really think about the fact that human beings are behind the firewalls, wireless networks, and Web applications they're attacking. They ignore that their actions often affect those human beings in negative ways, such as jeopardizing their job security.

On the flip side, odds are you have at least a handful of employees, contractors, interns, or consultants who intend to compromise sensitive information on your network for malicious purposes. These people don't hack in the way people normally suppose. Instead, they root around in files on server shares, delving into databases they know they shouldn't be in, sometimes stealing, modifying, and deleting sensitive information to which they have access. This behavior is often very hard to detect — especially given the widespread belief by management that users can and should be trusted to do the right things. This activity is perpetuated if these users passed their criminal background and credit checks before they were hired. Past behavior is often the best predictor of future behavior, but just because someone has a clean record and authorization to access sensitive systems doesn't mean he or she won't do anything bad. Criminals have to start somewhere!

As negative as breaking into computer systems often can be, hackers and malicious users play key roles in the advancement of technology. In a world without hackers, odds are that the latest intrusion prevention technology, data leakage protection, or vulnerability scanning tools would not exist. Such a world may not be bad, but technology does keep us in our jobs and keeps our field moving forward. Unfortunately, the technical security solutions can't ward off all malicious attacks and unauthorized use because hackers and (sometimes) malicious users are usually a few steps ahead of technology.

However you view the stereotypical hacker or malicious user, one thing is certain: Somebody will always try to take down your computer systems and compromise information by poking and prodding where he or she shouldn't, by all-out hacking, or by creating and launching automated worms and other malware. You must take the appropriate steps to protect your systems against this kind of intrusion.

Thinking like the bad guys

Malicious attackers often think and work just like thieves, kidnappers, and other organized criminals you hear about in the news every day. The smart ones constantly devise ways to fly under the radar and exploit even the smallest weaknesses that lead them to their target. The following are examples of how hackers and malicious users think and work. This list isn't intended to highlight specific exploits that I cover in this book or that I recommend you carry out, but rather to demonstrate the context and approach of a malicious mindset:

- *Evading an intrusion prevention system* by changing their MAC address and IP address every few minutes to get further into a network without being completely blocked
- *Exploiting a physical security weakness* by being aware of offices that have already been cleaned by the cleaning crew and are unoccupied (and thus easy to access with little chance of getting caught) by simply noting that the office blinds are opened and the curtains are pulled shut in the early morning
- *Bypassing Web access controls* by changing a malicious site's URL to its dotted decimal IP address equivalent and then converting it to hexadecimal for use in the Web browser
- *Using unauthorized software that would otherwise be blocked at the firewall* by changing the default TCP port that it runs on
- *Setting up a wireless "evil twin"* near a local Wi-Fi hotspot to lure unsuspecting Internet surfers onto a rogue network where their information can be captured and easily manipulated
- *Using a trusting colleague's user ID and password* to gain access to sensitive information that would otherwise be impossible to obtain
- *Unplugging the power cord or Ethernet connection to a networked CCTV security camera* that monitors access to the computer room or other sensitive areas and gaining unfettered access
- *Performing SQL injection or password cracking against a Web site* via a neighbor's unprotected wireless network in order to hide his or her identity

These people operate in countless ways and this list only presents a small number of the techniques hackers may use. Information security professionals need to think and work this way in order to really dig in and find security vulnerabilities that may not otherwise be uncovered.

Who Breaks into Computer Systems

Computer hackers have been around for decades. Since the Internet became widely used in the late 1990s, the mainstream public has started to hear more and more about hacking. Only a few hackers, such as John Draper (also known as Captain Crunch), and Kevin Mitnick, are really well known. Many more unknown hackers are looking to make a name for themselves. They're the ones you have to look out for.

In a world of black and white, describing the typical hacker is easy. A general stereotype of a hacker is an antisocial, pimply faced, teenage boy. But the world has many shades of gray and many types of hackers. Hackers are human and unique individuals, so an exact profile is hard to outline. The best broad description of hackers is that all hackers *aren't* equal. Each hacker has his or her own unique motives, methods, and skills. Hacker skill levels fall into three general categories:

- **Script kiddies:** These are computer novices who take advantage of the hacker tools, vulnerability scanners, and documentation available free on the Internet but don't have any knowledge of what's really going on behind the scenes. They know just enough to cause you headaches, but typically are very sloppy in their actions, leaving all sorts of digital fingerprints behind. Even though these guys are the stereotypical hackers that you hear about in the news media, they often need only minimal skills to carry out their attacks.
- **Criminal hackers:** These are skilled criminal experts who also write some of the hacking tools, including the scripts and other programs that the script kiddies and ethical hackers use. These folks also write such malware as viruses and worms. They can break into systems and cover their tracks. They can even make it look like someone else hacked their victims' systems.

  Advanced hackers are often very secretive and share information with their "subordinates" only when they are deemed worthy. Typically, for lower-ranked hackers to be considered worthy, they must possess some unique information or prove themselves through a high-profile hack. These hackers are arguably some of your worst enemies in information security. (Okay, maybe they're not as bad as untrained and careless users.) Fortunately, these elite hackers are not as plentiful as script kiddies.
- **Security researchers:** These are highly technical and publicly known IT professionals who not only monitor and track computer, network, and application vulnerabilities but also write the tools and other code to exploit them. If these guys didn't exist, we wouldn't have much in the way of open source security testing tools. I follow many of these security researchers on a weekly basis via their blogs, message boards, and articles and you should, too. Following the progress of these security researchers helps you stay up to date on both vulnerabilities and the latest and greatest security tools. I list the tools and related resources from various security researchers in Appendix A and throughout the book.

Regardless of age and complexion, hackers possess curiosity, bravado, and often very sharp minds.

Perhaps more important than a hacker's skill level is his or her motivation:

- **Hacktivists** try to disseminate political or social messages through their work. A hacktivist wants to raise public awareness of an issue. Examples of hacktivism are the Web sites that were defaced with the *Free Kevin* messages that promoted freeing Kevin Mitnick from prison for his famous hacking escapades. Others cases of hacktivism include messages about legalizing marijuana, protests against the war in Iraq, and many other social and political issues around the world. A more recent example was the use of Twitter to spread distributed denial of service (DDoS) instructions by students protesting the 2009 election results in Iran.
- **Cyberterrorists** attack government computers or public utility infrastructures, such as power grids and air-traffic control towers. They crash critical systems or steal classified government information. Countries take the threats these cyberterrorists pose so seriously that many mandate information-security controls in crucial industries, such as the power industry, to protect essential systems against these attacks.
- **Hackers for hire** are part of organized crime on the Internet. Not long ago, the Korean National Police Agency busted the Internet's largest known organized hacking ring, which had over 4,400 members. In another instance, police in the Philippines busted a multimillion-dollar organized hacking ring that sold cheap phone calls made through phone lines the ring had hacked into. Many of these hackers hire out themselves or their botnets for money — and lots of it!

These criminal hackers are in the minority, so don't think that you're up against millions of these villains. Many other hackers just love to tinker and only seek knowledge of how computer systems work. Your greatest threat works inside your building and has a valid network account, so don't discount the insider threat.

Why They Do It

The main reason hackers hack is because they can. Period. Okay, it goes a little deeper than that. Hacking is a casual hobby for some hackers — they hack just to see what they can and can't break into, usually testing only their own systems. These aren't the folks I write about in this book. I focus on those hackers who are obsessive about gaining notoriety or defeating computer systems, and those who have criminal intentions.

Many hackers get a kick out of outsmarting corporate and government IT and security administrators. They thrive on making headlines and being notorious cyberoutlaws. Defeating an entity or possessing knowledge that few other people have makes them feel better about themselves. Many of these hackers feed off the instant gratification of exploiting a computer system. They become obsessed with this feeling. Some hackers can't resist the adrenaline rush they get from breaking into someone else's systems. Often, the more difficult the job is, the greater the thrill is for hackers.

Hackers often promote individualism — or at least the decentralization of information — because many believe that all information should be free. They think cyberattacks are different from attacks in the real world. Hackers may easily ignore or misunderstand their victims and the consequences of hacking. Many hackers say they don't intend to harm or profit through their bad deeds, a belief that helps them justify their work. Many don't look for tangible payoffs. Just proving a point is often a sufficient reward for them.

The knowledge that malicious attackers gain and the self-esteem boost that comes from successful hacking might become an addiction and a way of life. Some attackers want to make your life miserable, and others simply want to be seen or heard. Some common motives are *revenge, basic bragging rights, curiosity, boredom, challenge, vandalism, theft for financial gain, sabotage, blackmail, extortion,* and *corporate espionage.* Hackers regularly cite these motives to explain their actions, but these motivations tend to be cited more commonly during difficult economic conditions.

Malicious users inside your network may be looking to gain information to help them with personal financial problems, to give them a leg up over a competitor, to seek revenge on their employers, to satisfy their curiosity, or to relieve boredom.

Many business owners and managers — even some network and security administrators — believe that they don't have anything that a hacker wants or that hackers can't do much damage if they break in. They couldn't be more mistaken. This dismissive kind of thinking about hacking helps support the bad guys and promote their objectives. Hackers can compromise a seemingly unimportant system to access the network and use it as a launching pad for attacks on other systems.

Remember that hackers often hack just because they can. Some hackers go for high-profile systems, but hacking into anyone's system helps them fit into hacker circles. Hackers exploit many people's false sense of security and go for almost any system they think they can compromise. Electronic information can be in more than one place at the same time, so if hackers merely copy information from the systems they break into, it's tough to prove that hackers possess that information.

Similarly, hackers know that a simple defaced Web page — however easily attacked — is not good for someone else's business. The following Web site shows a few examples of Web pages that have been defaced in the past: `http://zone-h.org/archive`.

Hacked sites can often persuade management and other nonbelievers to address information threats and vulnerabilities.

Computer breaches continue to get easier for several reasons:

- Widespread use of networks and Internet connectivity
- Anonymity provided by computer systems working over the Internet and often on the internal network (because, effectively, logging and especially log monitoring rarely takes place)
- Greater number and availability of hacking tools
- Large number of open wireless networks that help hackers cover their tracks
- Greater complexity and size of the codebase in the applications and databases being developed today
- Computer-savvy children
- Unlikelihood that attackers will be investigated or prosecuted if caught

Although most attacks go unnoticed or unreported, criminals who are discovered are often not pursued or prosecuted. When they're caught, hackers often rationalize their services as being altruistic and a benefit to society: They're merely pointing out vulnerabilities before someone else does. Regardless, if hackers are caught and prosecuted, the "fame and glory" reward system that hackers thrive on is threatened.

The same goes for malicious users. Typically, their shenanigans go unnoticed, but if they're caught, the security breach may be kept hush-hush in the name of shareholder value or not wanting to ruffle any feathers. However, recent information security and privacy laws and regulations are changing this because in most situations breach notification is required. Sometimes, the person is fired or asked to resign. Although public cases of internal breaches are becoming more common, these cases do not give a full picture of what's really taking place in the average organization.

Whether or not they want to, most executives now have to deal with all the state, federal, and international laws and regulations that require notifications of breaches or suspected breaches of sensitive information. This applies to external hacks, internal breaches, lost backup tapes, and more. Appendix A contains URLs to the sites giving information security and privacy laws and regulations that may affect your business.

Hacking in the name of liberty?

Many hackers exhibit behaviors that contradict their stated purposes — that is, they fight for civil liberties and want to be left alone, while at the same time, they love prying into other people's business. Many hackers call themselves civil libertarians and claim to support the principles of personal privacy and freedom. However, they contradict their words by intruding on the privacy and property of others. They often steal the property and violate the rights of others, but are willing to go to great lengths to get their own rights back from anyone who threatens them.

The case involving copyrighted materials and the Recording Industry Association of America (RIAA) is a classic example. Hackers have gone to great lengths to prove a point, from defacing the Web sites of organizations that support copyrights to illegally sharing music by using otherwise legal mediums like Kazaa, Gnutella, and Morpheus. Go figure.

Planning and Performing Attacks

Attack styles vary widely:

- **Some hackers prepare far in advance of a large attack.** They gather small bits of information and methodically carry out their hacks, as I outline in Chapter 4. These hackers are the most difficult to track.
- **Other hackers — usually the inexperienced script kiddies — act before they think through the consequences.** Such hackers may try, for example, to telnet directly into an organization's router without hiding their identities. Other hackers may try to launch a DoS attack against a Microsoft Exchange server without first determining the version of Exchange or the patches that are installed. These hackers usually are caught.
- **Malicious users are all over the map.** Some can be quite savvy based on their knowledge of the network and of how IT operates inside the organization. Others go poking and prodding around into systems they shouldn't be in — or shouldn't have had access to in the first place — and often do stupid things that lead security or network administrators back to them.

Although the hacker underground is a community, many of the hackers — especially advanced hackers — don't share information with the crowd. Most hackers do much of their work independently.

Hackers who network with one another use private message boards, anonymous e-mail addresses, hacker Web sites, and Internet Relay Chat (IRC). You can log on to many of these sites to see what hackers are doing.

Whatever approach they take, most malicious attackers prey on ignorance. They know the following aspects of real-world security:

- **The majority of computer systems aren't managed properly.** The computer systems aren't properly patched, hardened, and monitored. Attackers can often fly below the radar of the average firewall, an IPS, or access control system. This is especially true for malicious users whose actions are often not monitored at all, while, at the same time, they have full access to the very environment they can exploit.
- **Most network and security administrators simply can't keep up with the deluge of new vulnerabilities and attack methods.** These people often have too many tasks to stay on top of and too many other fires to put out. Network and security administrators may also fail to notice or respond because of poor time management and goal setting, but that's for another discussion.
- **Information systems grow more complex every year.** This is yet another reason why overburdened administrators find it difficult to know what's happening across the wire and on the hard drives of all their systems.

Time is an attacker's friend — and it's almost always on his or her side. By attacking through computers rather than in person, hackers have more control over the timing for their attacks:

- **Attacks can be carried out slowly, making them hard to detect.**
- **Attacks are frequently carried out after typical business hours** — often in the middle of the night, and from home, in the case of malicious users. Defenses are often weaker after hours — with less physical security and less intrusion monitoring — when the typical network administrator (or security guard) is sleeping.

If you want detailed information on how some hackers work or want to keep up with the latest hacker methods, several magazines are worth checking out:

- *2600 — The Hacker Quarterly* magazine (`www.2600.com`)
- *(IN)SECURE Magazine* (`www.net-security.org/insecuremag.php`)
- *Hackin9* (`http://hakin9.org`)
- *PHRACK* (`www.phrack.org/archives`)

Also, check out Lance Spitzner's Web site, `www.tracking-hackers.com`, for some great information on tracking hacker behavior.

Malicious attackers usually learn from their mistakes. Every mistake moves them one step closer to breaking into someone's system. They use this knowledge when carrying out future attacks. You, as an ethical hacker, need to do the same.

Maintaining Anonymity

Smart attackers want to remain as low-key as possible. Covering their tracks is a priority and many times, their success depends on them remaining unnoticed. They want to avoid raising suspicion so they can come back and access the systems in the future. Hackers often remain anonymous by using one of the following resources:

- Borrowed or stolen dial-up and VPN accounts from friends or previous employers
- Public computers at libraries, schools, or kiosks at the local mall
- Open wireless networks
- Internet proxy servers or anonymizer services
- Anonymous or disposable e-mail accounts from free e-mail services
- Open e-mail relays
- Unsecured computers — also called *zombies* or *bots* — at other organizations
- Workstations or servers on the victim's own network

If hackers use enough stepping stones for their attacks, they are hard to trace. Luckily, one of your biggest concerns — the malicious user — generally isn't quite as savvy. That is, unless the user is a network or security administrator.

Chapter 3

Developing Your Ethical Hacking Plan

In This Chapter

- Setting ethical hacking goals
- Selecting which systems to test
- Developing your ethical hacking testing standards
- Examining hacking tools

As an ethical hacker, you must plan your ethical hacking efforts before you start. A detailed plan doesn't mean that your testing must be elaborate. It just means that you're clear and concise about what to do. Given the seriousness of ethical hacking, make this as structured a process as possible.

Even if you only test a single Web application or workgroup of computers, establishing your goals, defining and documenting the scope of what you'll be testing, determining your testing standards, and gathering and familiarizing yourself with the proper tools for the task is critical. This chapter covers these steps to help you create a positive ethical hacking environment so you can set up for success.

Always make sure you have approval from management, executives, or your clients before you start implementing your ethical hacking plan.

Do you need insurance?

If you're an independent consultant or have a business with a team of ethical hackers, consider getting *professional liability insurance* (also known as *errors and omissions insurance*) from an insurance agent who specializes in business insurance coverage. This kind of insurance can be expensive, but will be well worth the expense if something goes awry and you need protection. Many customers even require the insurance before they'll hire you to do the work.

Establishing Your Goals

Your ethical hacking plan needs goals. The main goal of ethical hacking is to find vulnerabilities in your systems so you can make them more secure. You can then take this a step further:

- **Define more specific goals.** Align these goals with your business objectives. What are you and the management trying to get from this process?
- **Create a specific schedule with start and end dates as well as the times your testing is to take place.** These dates and times are critical components of your overall plan.

Before you begin any ethical hacking, you absolutely, positively need everything in writing and signed-off on. Document everything, and involve management in this process. Your best ally in your ethical hacking efforts is a manager who supports what you're doing.

The following questions can start the ball rolling when you define the goals for your ethical hacking plan:

- **Does ethical hacking support the mission of the business and its IT and security departments?**
- **What business goals are met by performing ethical hacking?** These goals may include the following:
  - Prepping for the internationally accepted security standard of ISO/IEC 27002:2005
  - Meeting federal regulations such as HIPAA, GLBA, or PCI DSS
  - Meeting contractual requirements of clients or business partners
  - Improving the company's image
- **How will ethical hacking improve security, IT, and the general business?**
- **What information are you protecting?** This could be personal health information, intellectual property, confidential client information, or private employee information.
- **How much money, time, and effort are you and your organization willing to spend on ethical hacking?**
- **What specific deliverables will there be?** *Deliverables* can include anything from high-level executive reports to detailed technical reports and write-ups on what you tested, along with the outcomes of your tests. You can deliver specific information that is gleaned during your testing, such as passwords and other confidential information.

- **What specific outcomes do you want?** Desired outcomes include the justification for hiring or outsourcing security personnel, increasing your security budget, meeting compliance requirements, or enhancing security systems.

After you know your goals, document the steps to get there. For example, if one goal is to develop a competitive advantage to keep existing clients and attract new ones, determine the answers to these questions:

- When will you start your ethical hacking?
- Will your ethical hacking be *blind*, in which you know nothing about the systems you're testing, or *knowledge based,* in which you're given specific information about the systems you're testing, such as IP addresses, host names, and even usernames and passwords?
- Will this testing be technical in nature, involve physical security assessments, or even use social engineering?
- Will you be part of a larger ethical hacking team, sometimes called a *tiger team* or *red team?*
- Will you notify your clients of what you're doing and when you're doing it? If so, how?

  Client notification is a critical issue. Many clients appreciate that you're taking steps to protect their information. Approach the testing in a positive way. Don't say, "We're breaking into your systems to see what information of yours is vulnerable to hackers," even if that's what you're doing. Instead, you can say that you're assessing the overall security of the client's systems so the information is as secure as possible.
- How will you know whether clients care about this?
- How will you notify clients that the organization is taking steps to enhance the security of their information?
- What measurements can ensure that these efforts are paying off?

Establishing your goals takes time, but you won't regret it. These goals are your road map. If you have any concerns, refer to these goals to make sure that you stay on track.

Determining Which Systems to Hack

You probably don't want — or need — to assess the security of all your systems at the same time. Assessing security of all your systems could be quite an undertaking and might lead to problems. I'm not recommending that you don't eventually assess every computer and application you have. I'm just

suggesting that whenever possible, you should break your ethical hacking projects into smaller chunks to make them more manageable. You might decide which systems to test based on a high-level risk analysis, answering questions such as

- What are your most critical systems? Which systems, if accessed without authorization, would cause the most trouble or suffer the greatest losses?
- Which systems appear most vulnerable to attack?
- Which systems are not documented, are rarely administered, or are the ones you know the least about?

After you've established your overall goals, decide which systems to test. This step helps you define a scope for your ethical hacking so that you establish everyone's expectations up front and better estimate the time and resources for the job.

The following list includes devices, systems, and applications that you may consider performing your hacking tests on:

- Routers and switches
- Firewalls
- Wireless access points and bridges
- Web, application, and database servers
- E-mail and file or print servers
- Workstations, laptops, and tablet PCs
- Mobile devices (such as PDAs and smart phones) that store confidential information
- Client and server operating systems

What specific systems you should test depends on several factors. If you have a small network, you can test everything. Consider testing just public-facing hosts such as e-mail and Web servers and their associated applications. The ethical hacking process is flexible. Base these decisions on what makes the most business sense.

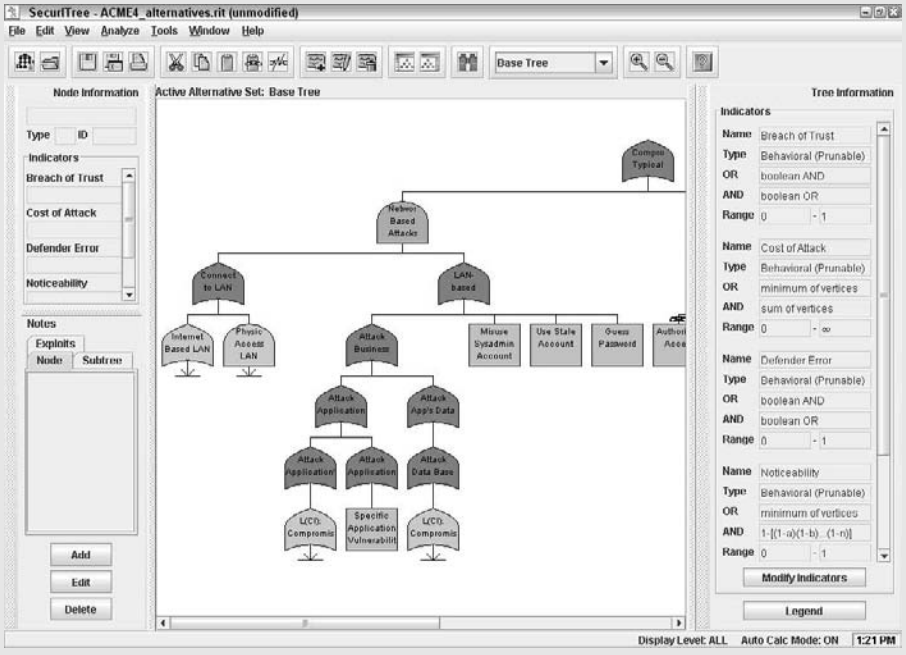
Start with the most vulnerable systems and consider the following factors:

- Where the computer or application resides on the network
- Which operating system and application(s) the system runs
- The amount or type of critical information stored on the system

Attack tree analysis

Attack tree analysis is the process of creating a flowchart-type mapping of how malicious attackers would attack a system. Attack trees are typically used in higher-level information risk analyses and by security-savvy development teams when planning out a new software project. If you really want to take your ethical hacking to the next level by thoroughly planning your attacks, working very methodically, and being more professional to boot, then attack tree analysis is just the tool you need.

The only drawback is that attack trees can take considerable time to draw out and require a fair amount of expertise. Why sweat it, though, when you can use a computer to do a lot of the work for you? A commercial tool called Secur*I*Tree, by Amenaza Technologies Limited (`www.amenaza.com`), specializes in attack tree analysis and should be in every serious security team's or professional's toolbox. The following figure shows a sample Secur*I*Tree attack tree analysis.



A previous security risk assessment, vulnerability test, or business impact analysis may already have generated this information. If so, that documentation can help identify systems for more testing.

Ethical hacking goes a few steps deeper than higher-level information risk assessments and vulnerability assessments. As an ethical hacker, you often start by gleaning information on all systems — including the organization as a whole — and then further assessing the most vulnerable systems. But again, this process is flexible. I discuss the ethical hacking methodology in Chapter 4.

Another factor that will help you decide where to start is to assess the systems that have the greatest visibility. For example, focusing on a database or file server that stores client or other critical information may make more sense — at least initially — than concentrating on a firewall or Web server that hosts marketing information about the company.

Creating Testing Standards

One miscommunication or slip-up can send the systems crashing during your ethical hacking tests. No one wants that to happen. To prevent mishaps, develop and document testing standards. These standards should include

- When the tests are performed, along with the overall timeline
- Which tests are performed
- How much knowledge of the systems you acquire in advance
- How the tests are performed, and from what source IP addresses (if performed across the Internet)
- What you do when a major vulnerability is discovered

This is a list of general best practices — you can apply more standards for your situation. The following sections describe these general best practices in more detail.

Timing

They say that it's "all in the timing." This is especially true when performing ethical hacking tests. Make sure that the tests you perform minimize disruption to business processes, information systems, and people. You want to avoid harmful situations such as miscommunicating the timing of tests and causing a DoS attack against a high-traffic e-commerce site in the middle of the day, or performing password-cracking tests in the middle of the night. It's amazing what a 12-hour time difference (2 p.m. during major production versus 2 a.m. during down time) can make when testing your systems! Everyone on the project needs to agree on a detailed timeline before you begin. Having the team members' agreement puts everyone on the same page and sets correct expectations.

If possible and practical, notify your Internet service providers (ISPs) or hosting/colo providers so that they're aware of the testing going on, to minimize the chance that they will block your traffic if they suspect malicious behavior that shows up on their firewall or intrusion detection system or intrusion prevention systems (IDS/IPS).

Your testing timeline should include specific short-term dates and times of each test, the start and end dates, and any specific milestones in between. You can develop and enter your timeline into a simple spreadsheet or Gantt chart, or you can include the timeline as part of your initial client proposal and contract. A timeline such as the following keeps things simple and provides a reference during testing:

Test Performed	*Tester*	*Start Time*	*Projected End Time*
Web application vulnerability scanning	Tommy Tinker	July 1, 6 a.m.	July 1, 10 a.m.
OS vulnerability exploitation	Amy Trusty	July 2, 12 p.m.	July 2, 5 p.m.

Specific tests

You might have been charged with performing a general *penetration test,* or you may want to perform specific tests, such as cracking passwords or trying to gain access to a Web application. Or you might be performing a social engineering test or assessing Windows on the network. However you test, you might not want to reveal the specifics of the testing. Even when your manager or client doesn't require detailed records of your tests, document what you are doing at a high level. Documenting your testing can help eliminate any potential miscommunication and keep you out of hot water.

Enabling logging on the systems you test provides evidence of what and when you test and more. You could even record screen actions using a tool such as Camtasia Studio (`www.camtasia.com`).

Sometimes, you might know the general tests that you perform, but if you use automated tools, it may be next to impossible to understand every test you perform completely. This is especially true when the software you're using receives real-time vulnerability updates from the vendor each time you run it. The potential for frequent updates underscores the importance of reading the documentation and readme files that come with the tools you use.

An updated program once bit me. I was performing an automated assessment on a client's Web site — the same test I performed the previous week. The client and I had scheduled the test date and time in advance. But I didn't know that the software vendor made some changes to its Web form submission tests, and I accidentally flooded the client's Web application, creating a DoS condition.

Luckily, this DoS condition occurred after business hours and didn't affect the client's operations. However, the client's Web application was coded to generate an alert e-mail for every form submission. The application developer and company's president received 4,000 e-mails in their inboxes within about 10 minutes — ouch! My experience is a perfect example of not knowing how my tool was configured by default and what it would do in this situation. I was lucky that the president was tech-savvy and understood the situation. Remember to have a contingency plan in case a situation like this occurs.

Blind versus knowledge assessments

Having some knowledge of the systems you're testing might be a good idea, but it's not required. But, a basic understanding of the systems you hack can protect you and others. Obtaining this knowledge shouldn't be difficult if you're hacking your own in-house systems. If you hack a client's systems, you might have to dig a little deeper into how the systems work so you're familiar with them. Doing so has always been my practice and I've never had a client ask for a full blind assessment because most people are scared of them. This doesn't mean that blind assessments aren't valuable, but the type of assessment you carry out depends on your specific needs.

The best approach is to plan on *unlimited* attacks, wherein any test is possible. The bad guys aren't hacking your systems within a limited scope, so why should you?

Consider whether the tests should be performed so that they're undetected by network administrators and any managed security service providers. While not required, this practice should be considered, especially for social engineering and physical security tests. I outline specific tests for those subjects in Chapters 5 and 6.

If too many insiders know about your testing, they might create a false sense of vigilance by improving their habits, which can end up negating the hard work you put into the testing. This doesn't mean you shouldn't tell anyone. *Always* have a main point of contact within the organization — preferably someone with decision-making authority — that both you and all employees can contact if and when something goes wrong with your testing.

Location

The tests you perform dictate where you must run them from. Your goal is to test your systems from locations accessible by malicious hackers. You can't predict whether you'll be attacked by a someone inside or outside your network, so cover all your bases. Combine external (public Internet) tests and internal (private network) tests.

You can perform some tests, like password cracking and network-infrastructure assessments, from your office. Having a true outsider who has no knowledge or vested interest perform other tests on routers, firewalls, and public Web applications might be a better idea.

For external hacks that require network connectivity, you might have to go off-site (a good excuse to work from home) or use an external proxy server. Better yet, if you can assign an available public IP address to your computer, simply plug in to the network on the outside of the firewall for a hacker's-eye view of your systems. Internal tests are easy because you need only physical access to the building and the network. You might be able to use a DSL line or cable modem already in place for visitors and similar users.

Reacting to vulnerabilities you find

Determine ahead of time whether you'll stop or keep going when you find a critical security hole. You don't need to keep hacking forever or until you crash all the systems. Just follow the path you're on until you can't hack it any longer (pardon the pun). When in doubt, the best thing to do is to have a specific goal in mind and then stop when that goal has been met.

Having said this, if you discover a major hole, I recommend contacting the right people as soon as possible so that they can begin fixing the issue right away. If you wait a few days or weeks, someone might exploit the vulnerability and cause damage that could've been prevented. As an employee, this could violate your employment agreement and even be considered negligence.

Silly assumptions

You've heard about what you make of yourself when you assume things. Even so, you make assumptions when you hack your systems. Here are some examples of those assumptions:

- Computers, networks, and people are available when you're testing.
- You have all the proper testing tools.
- The testing tools you use will minimize the chances of crashing the systems you test.
- You know all the risks of your tests.

Document all assumptions and have the management or your client sign off on them as part of your overall approval process.

Selecting Security Assessment Tools

Which security assessment tools you need depends on the tests you're going to run. While you can perform some ethical hacking tests with a pair of sneakers, a telephone, and a basic workstation on the network, but comprehensive testing is easier with hacking tools.

If you're not sure what tools to use, fear not. Throughout this book, I introduce a wide variety of tools — both free and commercial — that you can use to accomplish your tasks. Chapter 1 provides a list of commercial, freeware, and open source tools. Appendix A contains a comprehensive listing of tools for your reference.

It's important to know what each tool can and can't do and how to use each one. I suggest reading the manual and other help files. Unfortunately, some tools have limited documentation, which can be frustrating. You can search newsgroups and message boards and post a message if you're having trouble with a tool.

Hacking tools can be hazardous to your network's health. Be careful when you use them. Always make sure that you understand what every option does before you use it. Try your tools on test systems if you're not sure how to use them. These precautions help prevent DoS conditions and loss of data on your production systems.

You may despise some freeware and open source hacking tools. If these tools end up causing you more headaches than they're worth or don't do what you need them to do, consider purchasing commercial alternatives. They're often easier to use and typically generate better high-level executive reports. Some commercial tools are expensive, but their ease of use and functionality often justify the cost.

Chapter 4

Hacking Methodology

In This Chapter

- Examining steps for successful ethical hacking
- Gleaning information about your organization from the Internet
- Scanning your network
- Looking for vulnerabilities

Before you dive in head first with your ethical hacking, it's critical to have at least a basic methodology to work from. Ethical hacking involves more than just penetrating and patching a system or network. Proven techniques can help guide you along the hacking highway and ensure that you end up at the right destination. Using a methodology that supports your ethical hacking goals separates the professionals from the amateurs and helps ensure that you make the most of your time and effort.

Setting the Stage for Testing

In the past a lot of ethical hacking involved manual processes. Now, tools can automate various tasks. These tools allow you to focus on performing the tests and less on the specific steps involved. However, following a general methodology and understanding what's going on behind the scenes will help you.

Ethical hacking is similar to beta testing software. Think logically — like a programmer, a radiologist, or a home inspector — to dissect and interact with all the system components to see how they work. You gather information, often in many small pieces, and assemble the pieces of the puzzle. You start at point A with several goals in mind, run your tests (repeating many steps along the way), and move closer until you discover security vulnerabilities at point B.

The process used for ethical hacking is basically the same as the one a malicious attacker would use — the primary differences lie in the goals and how you achieve them. Another key difference is that you, as an ethical hacker, will eventually attempt to assess *all* your information systems for vulnerabilities and properly address them, rather than run a single exploit or attack a small number of systems. Today's attacks can come from any angle against any system, not just from the perimeter of your network and the Internet as you might have been taught in the past. Test every possible entry point, including partner, vendor, and client networks, as well as home users, wireless LANs, and laptop computers. Any human being, computer system, or physical component that protects your computer systems — both inside and outside your buildings — is fair game.

When you start rolling with your ethical hacking, keep a log of the tests you perform, the tools you use, the systems you test, and your results. This information can help you do the following:

- Track what worked in previous tests and why.
- Help prove that you didn't maliciously hack the systems.
- Correlate your testing with intrusion detection systems and other log files if trouble or questions arise.
- Document your final report.

In addition to taking general notes, taking screen captures of your results whenever possible is also helpful. These shots come in handy later should you need to show proof of what occurred, and they also will be useful as you generate your final report. Also, depending on the tools you use, these screen captures might be your only evidence of vulnerabilities or exploits when it comes time to write your final report. Chapter 3 lists the general steps involved in creating and documenting an ethical hacking plan.

Your main task is to simulate the information gathering and system compromises carried out by someone with malicious intent. This task can be a partial attack on one computer or it can constitute a comprehensive attack against the entire network. Generally, you look for weaknesses that malicious users and external attackers might exploit. You want to assess internal systems (processes and procedures that involve computers, networks, people, and physical infrastructures). Look for vulnerabilities; check how all your systems interconnect and how private systems and information are (or aren't) protected from untrusted elements.

These steps don't include specific information on the low-tech hacking methods that you use for social engineering and assessing physical security, but the techniques are basically the same. I cover these methods in more detail in Chapters 5 and 6.

If you're performing ethical hacking for a client, you may go the blind assessment route and start with just the company name and no other information. This blind assessment approach allows you to start from the ground up and gives you a better sense of the information and systems that malicious attackers can access publicly. However, keep in mind that this way of testing can take longer, and you may have an increased chance of missing some security vulnerabilities.

As an ethical hacker, you might not have to worry about covering your tracks or evading intrusion detection systems because everything you do is legitimate. But you might want to test systems stealthily. I discuss techniques that hackers use to conceal their actions in this book and outline some countermeasures for them, as well.

Seeing What Others See

Getting an outside look can turn up a ton of information about your organization and systems that others can see, through a process often called *footprinting.* Here's how to gather the information:

- Use a Web browser to search for information about your organization. Search engines, such as Google and Bing, are great places to start.
- Run network scans, probe open ports, and assess vulnerabilities to determine specific information about your systems. As an insider, you can use port scanners and Windows share-finder tools, such as GFI LANguard, to see what's accessible.

Whether you search generally or probe more technically, limit the amount of information you gather based on what's reasonable for you. You might spend an hour, a day, or a week gathering this information — how much time you spend depends on the size of the organization and the complexity of its information systems.

Gathering public information

The amount of information you can gather about an organization's business and information systems is staggering and widely available on the Internet. Your job is to find out what's out there. This information allows malicious attackers and employees to target specific areas of the organization, including departments and key individuals.

The following techniques can be used to gather information about your organization.

Web search

Performing a Web search or simply browsing your organization's Web site can turn up the following information:

- Employee names and contact info
- Important company dates
- Incorporation filings (for private companies)
- SEC filings (for public companies)
- Press releases about moves, organizational changes, and new products
- Mergers and acquisitions
- Patents and trademarks
- Presentations, articles, and Webcasts or Webinars

Microsoft is making headway into the search arena with Bing (`www.bing.com`). However, my favorite tool (and the favorite of many hackers) is still Google (`www.google.com`). This search engine ferrets out information — from word processing documents to graphics files — on any publicly accessible computer. And it's free. Entire books have been written about using Google, so expect any hacker (ethical or otherwise) to be very well versed on this useful tool. (See Chapter 14 for more about Google hacking.)

With Google, you can search the Internet in several ways:

- **By typing keywords:** This kind of search often reveals hundreds and sometimes millions of pages of information — such as files, phone numbers, and addresses — that you never guessed were available.
- **By performing advanced Web searches:** Google's advanced search options can find sites that link back to your company's Web site. This type of search often reveals a lot of information about partners, vendors, clients, and other affiliations.
- **By using switches to dig deeper into a Web site:** For example, if you want to find a certain word or file on your Web site, simply enter a line like one of the following into Google:

```
site:www.your_domain.com keyword
site:www.your_domain.com filename
```

You can even do a generic filetype search across the entire Internet to see what turns up, such as:

```
filetype:swf company_name
```

Use this search for Flash `.swf` files, which can be downloaded and decompiled to reveal sensitive information that can be used against your business, as I cover in detail in Chapter 14.

```
filetype:pdf company_name confidential
```

Use this search for PDF documents that might contain sensitive information that can be used against your business.

Web crawling

Web-crawling utilities, such as HTTrack Website Copier, can mirror your Web site by downloading every publicly accessible file from it. You can then inspect that copy of the Web site offline, digging into the following:

- The Web site layout and configuration
- Directories and files that might not otherwise be obvious or readily accessible
- The HTML and script source code of Web pages
- Comment fields

Comment fields often contain useful information such as names and e-mail addresses of the developers and internal IT personnel, server names, software versions, internal IP addressing schemes, and general comments about how the code works.

Web sites

The following Web sites may provide specific information about an organization and its employees:

- Government and business Web sites:
  - `www.hoovers.com` and `http://finance.yahoo.com` give detailed information about public companies.
  - `www.sec.gov/edgar.shtml` shows SEC filings of public companies.
  - `www.uspto.gov` offers patent and trademark registrations.
  - The Web site for your state's Secretary of State or similar organization can offer incorporation and corporate officer information.
- Background checks and other personal information:
  - ChoicePoint (`www.choicepoint.com`)
  - USSearch (`www.ussearch.com`)
  - ZabaSearch (`www.zabasearch.com`)

Mapping the network

When you map your network, you can search public databases and resources to see what other people know about your network.

Whois

The best starting point is to perform a Whois lookup by using any one of the Whois tools available on the Internet. You may have used Whois to check whether a particular Internet domain name is available.

For ethical hacking, Whois provides the following information that can give a hacker a leg up to start a social engineering attack or to scan a network:

- Internet domain name registration information, such as contact names, phone numbers, and mailing addresses
- DNS servers responsible for your domain

You can look up Whois information at one of the following places:

- Whois.net (`www.whois.net`)
- A domain registrar's site, such as `www.godaddy.com`
- Your ISP's tech support site

My favorite Whois tool is DNSstuff.com (`www.dnsstuff.com`). Although this tool is no longer free and is used to sell many services, it's still a good resource.

You can run DNS queries directly from the site to

- Display general domain-registration information
- Show which host handles e-mail (the Mail Exchanger or MX record) for a domain
- Map the location of specific hosts
- Determine whether the host is listed on certain spam blacklists

A free site you can use for more basic Internet domain queries is `www.dnstools.com`.

The following list shows various lookup sites for other categories:

- **Government:** `www.dotgov.gov`
- **Military:** `www.nic.mil`
- **AfriNIC:** `www.afrinic.net` (emerging Regional Internet Registry for Africa)

- **APNIC:** `www.apnic.net` (Regional Internet Registry for the Asia Pacific Region)
- **ARIN:** `https://ws.arin.net/whois/index.html` (Regional Internet Registry for North America, a portion of the Caribbean, and subequatorial Africa)
- **LACNIC:** `www.lacnic.net/en` (Latin American and Caribbean Internet Addresses Registry)
- **RIPE Network Coordination Centre:** `www.db.ripe.net/whois` (Europe, Central Asia, African countries north of the equator, and the Middle East)

If you're not sure where to look for a specific country, `https://www.arin.net/knowledge/rirs/countries.html` has a reference guide.

Google Groups

Google Groups (`http://groups.google.com`) can reveal surprising public network information. Search for such information as your fully qualified domain names (FQDNs), IP addresses, and usernames. You can search millions of Usenet posts that date back to 1981 for public and often very private information.

You might find some information that you didn't realize was made public, such as the following:

- A tech-support or message board post that divulges too much information about your systems. Many people who post messages like these don't realize that their messages are shared with the world or how long they are kept.
- Confidential company information posted by disgruntled employees or clients.

If you discover that confidential information about your company is posted online, you may be able to get it removed. Check out the Google Groups help page at `http://groups.google.com/support` for details.

Privacy policies

Check your Web site's privacy policy. A good practice is to let your site's users know what information is collected and how it's being protected, but nothing more.

Make sure that the people who write your privacy policies (often nontechnical lawyers or marketing managers) don't divulge details about your information security infrastructure. Be careful to avoid the example of an Internet start-up businessman who once contacted me about a business opportunity. During the conversation, he bragged about his company's security systems that ensured the privacy of client information (or so he thought). I went to his

Web site to check out his privacy policy. He had posted the brand and model of firewall he was using, along with other technical information about his network. This type of information could certainly be used against him by bad guys. Not a good idea.

Scanning Systems

Active information gathering produces more details about your network and helps you see your systems from an attacker's perspective. For instance, you can

- Use the information provided by your Whois searches to test other closely related IP addresses and hostnames. When you map out and gather information about a network, you see how its systems are laid out. This information includes determining IP addresses, hostnames (typically external but occasionally internal), running protocols, open ports, available shares, and running services and applications.
- Scan internal hosts when and where they are within the scope of your testing. (*Hint:* They really ought to be.) These hosts might not be visible to outsiders (at least you hope they're not), but you absolutely need to test them to see what rogue employees and other insiders can access. A worst-case situation is that the hacker has set up shop on the inside. Just to be safe, examine your internal systems for weaknesses.

If you're not completely comfortable scanning your systems, consider first using a lab with test systems or a system running virtual machine software, such as VMware Workstation or the open source alternative VirtualBox (`www.virtualbox.org`).

Hosts

Scan and document specific hosts that are accessible from the Internet and your internal network. Start by pinging either specific host names or IP addresses with one of these tools:

- The basic ping utility that's built in to your operating system
- A third-party utility that allows you to ping multiple addresses at the same time, such as SuperScan version 3 (`www.foundstone.com/us/resources/proddesc/superscan3.htm`) and NetScanTools Pro (`www.netscantools.com`) for Windows and fping (`www.fping.com`) for UNIX

The site `www.whatismyip.com` shows how your gateway IP address appears on the Internet. Just browse to that site, and your public IP address (your firewall or router — preferably not your local computer) appears. This information gives you an idea of the outermost IP address that the world sees.

Open ports

Scan for open ports by using network scanning tools:

- Scan network ports with SuperScan or Nmap (`http://nmap.org`). See Chapter 8 for details.
- Listen to network traffic with a network analyzer, such as OmniPeek (`www.wildpackets.com`) and Wireshark (`www.wireshark.com`). I cover this topic in various chapters throughout this book.

Scanning *internally* is easy. Simply connect your PC to the network, load the software, and fire away. Scanning from *outside* your network takes a few more steps, but it can be done. The easiest way to connect and get an "outside-in" perspective is to assign your computer a public IP address and plug that workstation into a switch or hub on the public side of your firewall or router. Physically, the computer is not on the Internet looking in, but this type of connection works just the same as long as it's outside your firewall and router. You can also do this outside-in scan from home or a remote office location.

Determining What's Running on Open Ports

As an ethical hacker, you should glean as much information as possible after scanning your systems. You can often identify the following information:

- Protocols in use, such as IP, IPX, and NetBIOS
- Services running on the hosts, such as e-mail, Web servers, and database applications
- Available remote access services, such as Windows Terminal Services/Remote Desktop, VNC, and Secure Shell (SSH)
- VPN services, such as PPTP, SSL, and IPSec
- Required authentication for network shares

You can look for the following sampling of open ports (your network-scanning program reports these as accessible or open):

- Ping (ICMP echo) replies; ICMP traffic is allowed to and from the host
- TCP port 21, showing that FTP is running
- TCP port 23, showing that telnet is running
- TCP ports 25 or 465 (SMTP and SMPTS), 110 or 995 (POP3 and POP3S), or 143 or 993 (IMAP and IMAPS), showing that an e-mail server is running

- TCP/UDP port 53, showing that a DNS server is running
- TCP ports 80, 443, and 8080, showing that a Web server or Web proxy server is running
- TCP/UDP ports 135, 137, 138, 139 and, especially, 445, showing that an unprotected Windows host is running

Thousands of ports can be open — 65,536 each for both TCP and UDP, to be exact. I cover many popular port numbers when describing hacks throughout this book. A continually updated listing of all well-known port numbers (ports 0–1023) and registered port numbers (ports 1024–49151), with their associated protocols and services, is located at `www.iana.org/assignments/port-numbers`. You can also perform a port-number lookup at `www.cotse.com/cgi-bin/port.cgi`.

If you detect a Web server running on the system that you test, you can check the software version by using one of the following methods:

- Type the site's name followed by a page that you know doesn't exist, such as *`www.your_domain.com`*`/1234.html`. Many Web servers return an error page showing detailed version information.
- Use Netcraft's *What's that site running?* search utility (`www.netcraft.com`), which connects to your server from the Internet and displays the Web server version and operating system, as shown in Figure 4-1.

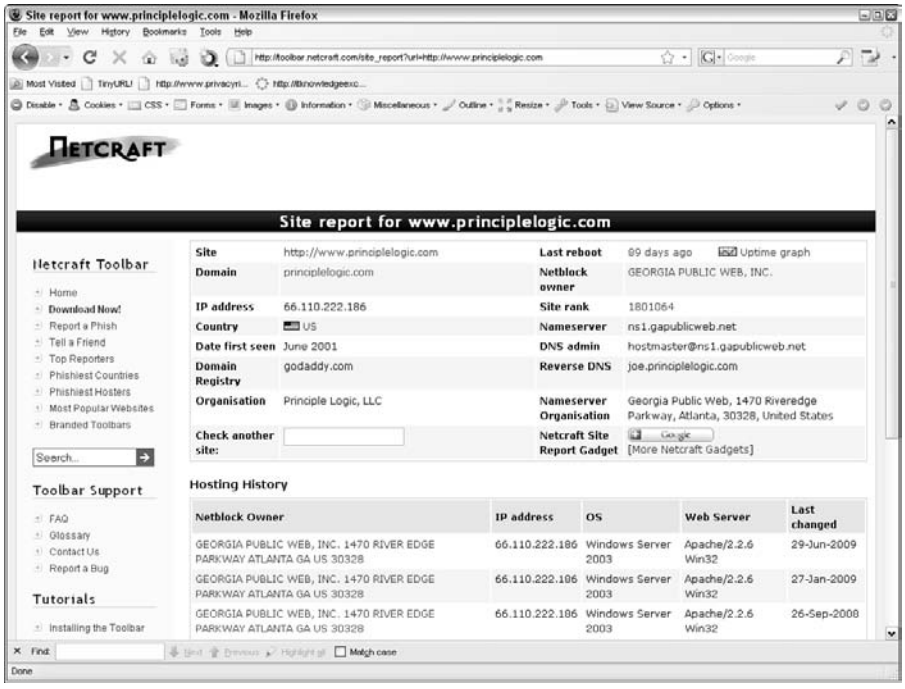


Figure 4-1: Netcraft's Web server version utility.

You can dig deeper for more specific information on your hosts:

- NMapWin (`http://sourceforge.net/projects/nmapwin`) can determine the system OS version.
- An enumeration utility (such as DumpSec at `www.systemtools.com/somarsoft/?somarsoft.com`) can extract users, groups, and file and share permissions directly from Windows.
- Many systems return useful banner information when you connect to a service or application running on a port. For example, if you telnet to an e-mail server on port 25 by entering `telnet mail.your_domain.com 25` at a command prompt, you may see something like this:

  ```
  220 mail.your_domain.com ESMTP all_the_version_info_
  you_need_to_hack Ready
  ```

  Most e-mail servers return detailed information, such as the version and the current service pack installed. After you have this information, you (and the bad guys) can determine the vulnerabilities of the system from some of the Web sites listed in the next section.
- A share-finder tool, such as the one built in to GFI LANguard, can find open Windows shares.
- An e-mail to an invalid address might return with detailed e-mail header information. A bounced message often discloses information that can be used against you, including internal IP addresses and software versions. On certain Windows systems, you can use this information to establish unauthenticated connections and sometimes even map drives. I cover these issues in Chapter 13.

Assessing Vulnerabilities

After finding potential security holes, the next step is to confirm whether they are vulnerabilities in your system or network. Before you test, perform some manual searching. You can research hacker message boards, Web sites, and vulnerability databases, such as these:

- Common Vulnerabilities and Exposures (`http://cve.mitre.org/cve`)
- US-CERT Vulnerability Notes Database (`www.kb.cert.org/vuls`)
- NIST National Vulnerability Database (`http://nvd.nist.gov`)

These sites list known vulnerabilities — at least the formally classified ones. As I cover in this book, you see that many other vulnerabilities are more generic in nature and can't easily be classified. If you can't find a vulnerability documented on one of these sites, search the vendor's site. You can also find a list of commonly exploited vulnerabilities at `www.sans.org/top20`. This

site contains the SANS Top 20 Vulnerabilities consensus list, which is compiled and updated by the SANS organization.

If you don't want to research your potential vulnerabilities and can jump right into testing, you have a couple of options:

- **Manual assessment:** You can assess the potential vulnerabilities by connecting to the ports that are exposing the service or application and poking around in these ports. You should manually assess certain systems (such as Web applications). The vulnerability reports in the preceding databases often disclose how to do this — at least generally. If you have a lot of free time, performing these tests manually might work for you.
- **Automated assessment:** Manual assessments are a great way to learn, but people usually don't have the time for most manual steps. If you're like me, you scan for vulnerabilities automatically when you can.

Many great vulnerability assessment tools test for vulnerabilities on specific platforms (such as Windows and UNIX) and types of networks (either wired or wireless). They test for specific system vulnerabilities and some even focus on the SANS Top 20 list. Versions of these tools can map the business logic within a Web application; others can help software developers test for code flaws. The drawback to these tools is that they find only individual vulnerabilities; they often don't correlate vulnerabilities across an entire network. However, the advent of event-correlation and vulnerability management applications is allowing these tools to correlate these vulnerabilities.

One of my favorite ethical hacking tools is a vulnerability scanner called QualysGuard Suite by Qualys (`www.qualys.com`). It's both a port scanner and vulnerability assessment tool, and it offers a great deal of help for vulnerability management. You don't even need a computer to run it because QualysGuard is a Software as a Service (SaaS) commercial tool. Just browse to the Qualys Web site, log in to your account, and enter the IP address of the systems you want to test. Qualys also has an appliance that you can install on your network that allows you to scan internal systems. You simply schedule the assessment, and then the system runs tests and generates excellent reports, such as these:

- An executive report containing general information from the results of the scan, as shown in Figure 4-2.
- A technical report of detailed explanations of the vulnerabilities and specific countermeasures.

Like most good security tools, you pay for QualysGuard — it isn't the *least* expensive tool, but you get what you pay for. With QualysGuard, you buy a block of scans based on the number of scans you run.

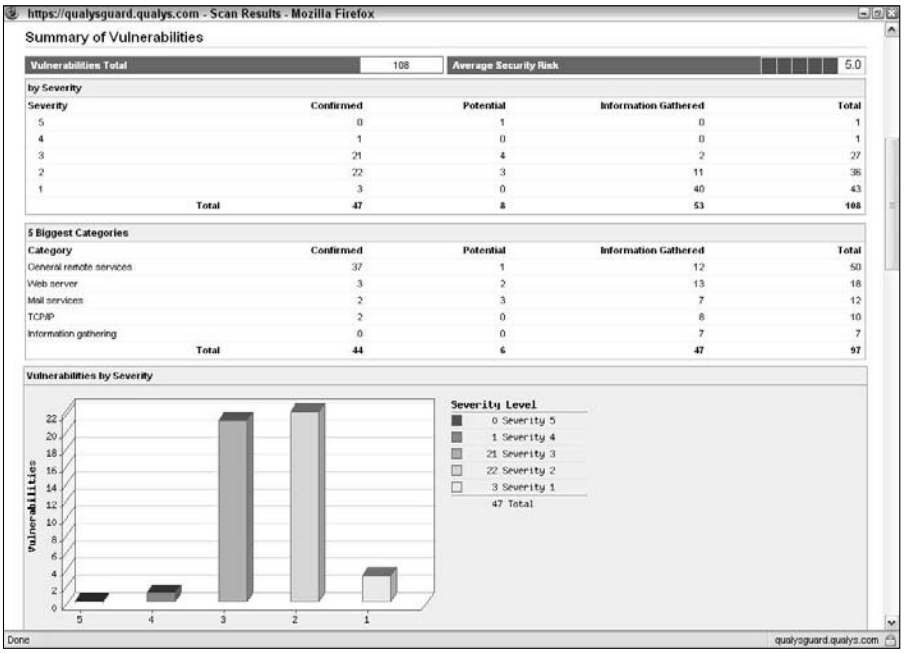


Figure 4-2: Executive summary data in a QualysGuard vulnerability assessment report.

Assessing vulnerabilities with a tool like QualysGuard requires follow-up expertise. You can't rely on the scan results alone. You have to validate the vulnerabilities it reports. Study the reports to base your recommendations on the context and criticality of the tested systems.

Penetrating the System

You can use identified critical security holes to do the following:

- Gain further information about the host and its data.
- Obtain a remote command prompt.
- Start or stop certain services or applications.
- Access other systems.
- Disable logging or other security controls.
- Capture screen shots.
- Access sensitive files.
- Send an e-mail as the administrator.

- Perform SQL injection attacks.
- Launch another type of DoS attack.
- Upload a file proving your victory.

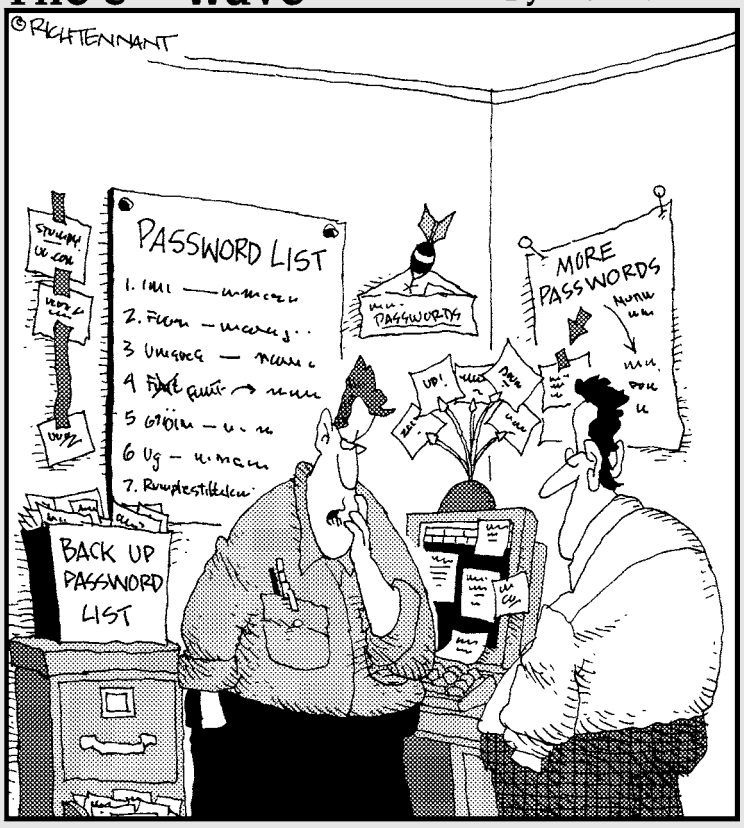
Metasploit (`www.metasploit.com/framework`) is great for exploiting many of the vulnerabilities you find and allows you to obtain complete system penetration. Ideally, you've already made your decision on whether to fully exploit the vulnerabilities you find. You might want to leave well enough alone by just demonstrating the existence of the vulnerabilities and not actually exploiting them.

If you want to delve into the methodology component even further, I recommend you check out the Open Source Security Testing Methodology Manual (`www.isecom.org/osstmm`) for more information.

Part II

Putting Ethical Hacking in Motion

The 5th Wave By Rich Tennant



"Well, whoever stole my passwords was sure clever. Especially since none of my reminders are missing."

In this part . . .

Let the games begin! You've waited long enough — now's the time to start testing your systems. But where do you start? How about with your three Ps — your people, your physical systems, and your passwords? These are, after all, three of the most easily and commonly attacked targets in your organization.

This part starts with a discussion of hacking people (as opposed to hacking up people; this is *social engineering,* not *The Texas Chainsaw Massacre*). It then goes on to look at physical security vulnerabilities. Of course, I'd be remiss in a part about people if I skipped passwords, so I cover the technical details of testing those as well. This is a great way to get the ball rolling to warm you up for the more specific hacks later in the book.

Chapter 5

Social Engineering

In This Chapter

- Introducing social engineering
- Examining the ramifications of social engineering
- Understanding and using social engineering techniques
- Protecting your organization against social engineering

Social engineering takes advantage of the weakest link in any organization's information security defenses: people. Social engineering is "people hacking" and involves maliciously exploiting the trusting nature of human beings to obtain information that can be used for personal gain.

Social engineering is one of the toughest hacks to perpetrate because it takes great skill to come across as trustworthy to a stranger. It's also by far the toughest hack to protect against because people are involved. In this chapter, I explore the ramifications of social engineering, techniques for your own ethical hacking efforts, and specific countermeasures to defend against social engineering.

Social Engineering 101

Typically, malicious attackers pose as someone else to gain information they couldn't access otherwise. They then take the information they obtain from their victims and wreak havoc on network resources, steal or delete files, and even commit industrial espionage or some other form of fraud against the organization they attack. Social engineering is different from *physical security* exploits, such as shoulder surfing and dumpster diving, but they are related and often are used in tandem.

Here are some examples of social engineering:

- **False support personnel** claim that they need to install a patch or new version of software on a user's computer, talk the user into downloading the software, and obtain remote control of the system.
- **False vendors** claim to need to update the organization's accounting package or phone system, ask for the administrator password, and obtain full access.
- **Phishing e-mails** sent by external attackers gather user IDs and passwords of unsuspecting recipients. The bad guys then use those passwords to gain access to bank accounts and more. A related attack exploits cross-site scripting on Web forms. I cover Web application security in detail in Chapter 14.
- **False employees** notify the security desk that they have lost their keys to the computer room, receive a set of keys from security, and obtain unauthorized access to physical and electronic information.

Sometimes, social engineers act as forceful and knowledgeable employees, such as managers or executives. At other times they might play the roles of extremely uninformed or naïve employees. They also might pose as outsiders, such as IT consultants or maintenance people. Social engineers often switch from one mode to the other, depending on the people they speak to.

Effective information security — especially the security required for fighting social engineering — begins and ends with your users. Other chapters in this book provide great technical advice, but never forget that basic human communications and interaction also affect the level of security. The *candy-security* adage is "Hard, crunchy outside; soft, chewy inside." The *hard, crunchy outside* is the layer of mechanisms — such as firewalls, intrusion detection systems, and encryption — that organizations rely on to secure their information. The *soft, chewy inside* is the people and the systems inside the organization. If the bad guys can get past the thick outer layer, they can compromise the (mostly) defenseless inner layer.

Before You Start

I approach the ethical hacking methodologies in this chapter differently than in subsequent chapters. Social engineering is an art and a science. Social engineering takes great skill to perform as an ethical hacker and depends upon your personality and overall knowledge of the organization you test. If social engineering isn't natural for you, consider using the information in this chapter for educational purposes — at least at first — until you have more time to study the subject. Don't hesitate to hire a third party to perform this testing if that makes the best business sense for now.

A case study in social engineering with Ira Winkler

In this case study, Ira Winkler, a world-renowned social engineer, graciously shared an interesting study in social engineering.

The Situation

Mr. Winkler's client wanted a general gauge of the organization's security awareness level. Ira and his accomplice went for the pot of gold and tested the organization's susceptibility to social engineering. To start, they scoped out the main entrance of the client's building and found that the reception area and security desk was in the middle of a large lobby and was staffed by a receptionist. The next day, the two men walked into the building during the morning rush while pretending to talk on cell phones. They stayed at least 15 feet from the attendant and simply ignored her as they walked by.

After they were inside the facility, they found a conference room to set up shop in. They sat down to plan the rest of the day and decided a facility badge would be a great start. Mr. Winkler called the main information number and asked for the office that makes the badges. He was forwarded to the reception/security desk. Ira then pretended to be the CIO and told the person on the other end of the line that he wanted badges for a couple of subcontractors. The person responded, "Send the subcontractors down to the main lobby."

When Mr. Winkler and his accomplice arrived, a uniformed guard asked what they were working on, and they mentioned computers. The guard then asked them if they needed access to the computer room! Of course, they said, "That would help." Within minutes, they both had badges with access to all office areas and the computer operations center. They went to the basement and used their badges to open the main computer room door. They walked right in and were able to access a Windows server, load the user administration tool, add a new user to the domain, and make the user a member of the administrators' group. Then they quickly left.

The two men had access to the entire corporate network with administrative rights within two hours. They also used the badges to perform after-hours walkthroughs of the building. While doing so, they found the key to the CEO's office and planted a mock bug there.

The Outcome

Nobody outside the team knew what the two men had done until they were told after the fact. After the employees were informed, the guard supervisor called Mr. Winkler and wanted to know who issued the badges. Mr. Winkler informed him that the fact that the security office didn't know who issued the badges was a problem in and of itself, and that he does not disclose that information.

How This Could Have Been Prevented

According to Mr. Winkler, the security desk should be located closer to the entrance, and the company should have a formal process for issuing badges. Access to special areas like the computer room should require approval from a known entity, as well. After access is granted, a confirmation should be sent to the approver. Also, the server screen should be locked and the Windows account should not be logged on unattended. Any addition of an administrator-level account should be audited and appropriate parties should be alerted.

Ira Winkler, CISSP, CISM, is founder and president of the Internet Security Advisors Group. You can find more of his case studies in his book *Spies Among Us: How to Stop the Spies, Terrorists, Hackers, and Criminals You Don't Even Know You Encounter Every Day* (Wiley).

You can use the information in this chapter to perform specific tests or improve information security awareness in your organization. Social engineering can harm people's jobs and reputations, and confidential information could be leaked. Proceed with caution, and think before you act.

You can perform social engineering attacks in millions of ways. For this reason, and because training specific behaviors in a single chapter is next to impossible, I don't provide how-to instructions for carrying out social engineering attacks. Instead, I describe specific social engineering scenarios that have worked for other hackers — both ethical and unethical. You can tailor these same tricks and techniques to your specific situation.

An outsider to the organization might perform these social engineering techniques best. If you perform these tests against your organization, acting as an outsider might be difficult if everyone knows you. This risk of recognition might not be a problem in larger organizations, but if you have a small, close-knit company, people might catch on to your antics.

You can outsource social engineering testing to a trusted consulting firm or even have a trusted colleague perform the tests for you. The key word here is *trusted*. If you involve someone else, you must get references, perform background checks, and have the testing approved by management in writing beforehand. I cover the topic of outsourcing security and ethical hacking in Chapter 18.

Why Attackers Use Social Engineering

Many bad guys use social engineering to break into systems simply because they can. They want someone to open the door to the organization so that they don't have to break in and risk being caught. Firewalls, access controls, and authentication devices can't stop a determined social engineer.

Most social engineers perform their attacks slowly, to avoid suspicion. Social engineers gather bits of information over time and use the information to create a broader picture. Alternatively, some social engineering attacks can be performed with a quick phone call or e-mail. The methods used depend on the attacker's style and abilities.

Social engineers know that many organizations don't have formal data classifications, access-control systems, incident response plans, and security awareness programs, and they often take advantage of these weaknesses.

Social engineers often know a little about a lot of things — both inside and outside their target organizations — because this knowledge helps them in their efforts. The more information social engineers gain about organizations,

the easier it is for them to pose as employees or other trusted insiders. Social engineers' knowledge and determination give them the upper hand over average employees who don't recognize the value of the information social engineers seek.

Understanding the Implications

Many organizations have enemies who want to cause trouble through social engineering. These enemies might be current or former employees seeking revenge, competitors wanting a leg up, or hackers trying to prove their skills.

Regardless of who causes the trouble, every organization is at risk — especially with the Web, which can help facilitate hacking and information gathering. Larger companies spread across several locations are often more vulnerable, but smaller companies might also be attacked. Everyone, from receptionists to security guards to IT personnel, is a potential victim of social engineering. Help desk and call center employees are especially vulnerable because they are trained to be helpful and forthcoming with information. Even the average, untrained user is susceptible to attack.

Social engineering has serious consequences. Because the objective of social engineering is to coerce someone for information to lead to ill-gotten gain, anything is possible. Effective social engineers can obtain the following information:

- User or administrator passwords
- Security badges or keys to the building and even to the computer room
- Intellectual property such as design specifications, formulae, or other research and development documentation
- Confidential financial reports
- Private and confidential employee information
- Customer lists and sales prospects

If any of the preceding information is leaked, financial losses, lower employee morale, decreased customer loyalty, and even legal and regulatory compliance issues can result. The possibilities are endless.

Another reason that social engineering attacks are difficult to protect against is because they aren't well documented. Because so many possible methods exist, recovery and protection are difficult after the attack. The *hard, crunchy outside* of firewalls and intrusion detection systems often creates a false sense of security, making the problem even worse.

With social engineering, you never know the next method of attack. The best things you can do are to remain vigilant, understand the social engineer's methodology, and protect against the most common attacks through ongoing security awareness in your organization. I discuss how you can do this in the rest of this chapter.

Performing Social Engineering Attacks

The process of social engineering is actually pretty basic. Generally, social engineers discover the details of organizational processes and information systems to perform their attacks. With this information, they know what to pursue. Hackers typically perform social engineering attacks in four simple steps:

1. Perform research.
2. Build trust.
3. Exploit relationships for information through words, actions, or technology.
4. Use the information gathered for malicious purposes.

These steps can include numerous substeps and techniques, depending on the attack being performed.

Before social engineers perform their attacks, they need a goal. This is the first step in these attackers' processes for social engineering, and this goal is most likely already implanted in their minds. What do they want to accomplish? What are the social engineers trying to hack? Why? Do they want intellectual property, server passwords, or security badges, or do they simply want to prove that the company's defenses can be penetrated? In your efforts as an ethical hacker performing social engineering, determine this overall goal before you move forward.

Phishing for information

After social engineers have a goal in mind, they typically start the attack by gathering public information about their victim(s). Many social engineers acquire information slowly over time so they don't raise suspicion. Obvious information gathering is a tip-off when defending against social engineering. I mention other warning signs to be aware of throughout the rest of this chapter.

Regardless of the initial research method, all a hacker needs to penetrate an organization is an employee list, a few key internal phone numbers, or a company calendar.

Using the Internet

Today's basic research medium is the Internet. A few minutes searching on Google or other search engines, using simple keywords, such as the company name or specific employees' names, often produces a lot of information. You can find even more information in SEC filings at `www.sec.gov` and at such sites as `www.hoovers.com` and `http://finance.yahoo.com`. (Many organizations — especially their management — would be dismayed to discover the organizational information that is available online.) By using this search-engine information and browsing the company's Web site, the attacker often has enough information to start a social engineering attack.

The bad guys can pay just a few dollars for a comprehensive online background check on individuals. These searches can turn up practically any public — and sometimes private — information about a person in minutes.

Dumpster diving

Dumpster diving is a little more risky — and it's certainly messy, but it's a highly effective method of obtaining information. This method involves literally rummaging through trash cans for information about a company.

Dumpster diving can turn up even the most confidential information because many employees assume that their information is safe after it goes into the trash. Most people don't think about the potential value of the paper they throw away. These documents often contain a wealth of information that can tip off the social engineer with information needed to penetrate the organization further. The astute social engineer looks for the following printed documents:

- Internal phone lists
- Organizational charts
- Employee handbooks, which often contain security policies
- Network diagrams
- Password lists
- Meeting notes
- Spreadsheets and reports
- Printouts of e-mails that contain confidential information

Shredding documents is effective only if the paper is *cross-shredded* into tiny pieces of confetti. Inexpensive shredders that shred documents only in long strips are basically worthless against a determined social engineer. With a little time and tape, a social engineer can easily piece a document back together.

Hackers often gather confidential personal and business information from others by listening in on conversations held in restaurants, coffee shops, and airports. People who speak loudly when talking on their cell phones are also a great source of sensitive information for social engineers. (Poetic justice, perhaps?) While writing in public places and eating in restaurants, it's amazing what I hear others divulge without even trying to listen.

The bad guys also look in the trash for CD-ROMs and DVDs, old computer cases (especially those with hard drives still intact), and backup tapes.

See Chapter 6 for more on trash and other physical security issues, including countermeasures for protecting against dumpster divers.

Phone systems

Attackers can obtain information by using the dial-by-name feature built in to most voicemail systems. To access this feature, you usually just press 0 after calling the company's main number or after you enter someone's voice mailbox. This trick works best after hours to ensure no one answers.

Attackers can protect their identities if they can hide where they call from. Here are some ways they can hide their locations:

- **Residential phones** sometimes can hide their numbers from caller ID by dialing *67 before the phone number.

  This feature isn't effective when calling toll-free numbers (800, 888, 877, 866) or 911.
- **Business phones** in an office using a phone switch are more difficult to spoof. However, all the attacker usually needs is the user guide and administrator password for the phone switch software. In many switches, the attacker can enter the source number — including a falsified number, such as the victim's home phone number. Voice over Internet Protocol (VoIP) phone systems are making this a non-issue, however.
- **VoIP Servers** such as the open source Asterisk (`www.asterisk.org`) can be used and configured to send any number they want.

Social engineers can find interesting bits of information, at times, such as when their victims are out of town, just by listening to voicemail messages. They can even study victims' voices by listening to their voicemail messages, podcasts, or Webcasts so they can learn to impersonate those people.

Building trust

Trust — so hard to gain, yet so easy to lose. Trust is the essence of social engineering. Most people trust others until a situation forces them not to. People want to help one another, especially if trust can be built and the

request for help seems reasonable. Most people want to be team players in the workplace and don't realize what can happen if they divulge too much information to a "trusted" source. This trust allows social engineers to accomplish their goals. Of course, building deep trust often takes time. Crafty social engineers can gain it within minutes or hours. How do they do it?

- **Likability:** Who can't relate to a nice person? Everyone loves courtesy. The friendlier social engineers are — without going overboard — the better their chances of getting what they want. Social engineers often begin to build a relationship by establishing common interests. They often use the information they gain in the research phase to determine what the victim likes and to pretend that they like those things, too. They can phone victims or meet them in person and, based on information the social engineers have discovered about the person, start talking about local sports teams or how wonderful it is to be single again. A few low-key and well-articulated comments can be the start of a nice new relationship. Of course, good looks don't hurt either.
- **Believability:** Believability is based in part on the knowledge that social engineers have and how likable they are. Social engineers also use impersonation — perhaps by posing as new employees or fellow employees that the victim hasn't met. They may even pose as vendors who do business with the organization. They often modestly claim authority to influence people. The most common social engineering trick is to do something nice so that the victim feels obligated to be nice in return or to be a team player for the organization.

Exploiting the relationship

After social engineers obtain the trust of their unsuspecting victims, they coax them into divulging more information than they should. Whammo — they can go in for the kill. Social engineers do this through face-to-face or electronic communication that victims feel comfortable with, or they use technology to get victims to divulge information.

Deceit through words and actions

Wily social engineers can get inside information from their victims in many ways. They are often articulate and focus on keeping their conversations moving without giving their victims much time to think about what they're saying. However, if they're careless or overly anxious during their social engineering attacks, the following tip-offs might give them away:

- Acting overly friendly or eager
- Mentioning names of prominent people within the organization
- Bragging about authority within the organization
- Threatening reprimands if requests aren't honored

- Acting nervous when questioned (pursing the lips and fidgeting — especially the hands and feet, because controlling body parts that are farther from the face require more conscious effort)
- Overemphasizing details
- Experiencing physiological changes, such as dilated pupils or changes in voice pitch
- Appearing rushed
- Refusing to give information
- Volunteering information and answering unasked questions
- Knowing information that an outsider should not have
- Using insider speech or slang as a known outsider
- Asking strange questions
- Misspelling words in written communications

A good social engineer isn't obvious with the preceding actions, but these are some of the signs that malicious behavior is in the works.

Social engineers often do a favor for someone and then turn around and ask that person if he or she would mind helping them. This common social engineering trick works pretty well. Social engineers also often use what's called *reverse social engineering.* This is where they offer help if a specific problem arises; some time passes, the problem occurs (often by their doing), and then they help fix the problem. They may come across as heroes, which can further their cause. Social engineers might ask an unsuspecting employee for a favor. Yes — they just outright ask for a favor. Many people fall for this trap.

Impersonating an employee is easy. Social engineers can wear a similar-looking uniform, make a fake ID badge, or simply dress like the real employees. People think, "Hey — he looks and acts like me, so he must be one of us." Social engineers also pretend to be employees calling from an outside phone line. This trick is an especially popular way of exploiting help desk and call center personnel. Social engineers know that these employees fall into a rut easily because their tasks are repetitive, such as saying, "Hello, can I get your customer number, please?"

Deceit through technology

Technology can make things easier — and more fun — for the social engineer. Often, a malicious request for information comes from a computer or other electronic entity the victims think they can identify. But spoofing a computer name, an e-mail address, a fax number, or a network address is easy. Fortunately, you can take a few countermeasures against this type of attack, as described in the next section.

Even professionals can be socially engineered

Here's how I fell prey to a social engineer because I didn't think before I spoke. One day, I was having trouble with my high-speed Internet connection. I figured I could just use dial-up access because it's better than nothing for e-mail and other basic tasks. I contacted my ISP and told the tech-support guy that I couldn't remember my password. This sounds like the beginning of a social engineering stunt that I could've pulled off, but *I* got taken instead. The slick tech-support guy paused for a minute, as if he were pulling up my account info, and then asked, "What password did you try?"

Stupid me, I proceeded to repeat all the passwords it could've been. The phone went quiet for a moment. He reset my password and told me what it was. After I hung up, I thought, "What just happened? I just got social engineered!" It may not have been intentional on his part, but it was a dumb mistake on my part. Man, was I mad at myself. I changed all the passwords that I divulged related to my Internet account in case he used that information against me. I still bet to this day that he was just experimenting with me. Lesson learned: Never, under any circumstances, divulge your password to someone else — another employee, your boss, whomever — even if they ask for it. The consequences just aren't worth the perceived benefit.

Hackers can deceive through technology by sending e-mail that asks victims for critical information. Such an e-mail usually provides a link that directs victims to a professional- and legitimate-looking Web site that "updates" such account information as user IDs, passwords, and Social Security numbers. They might also do this on social networking sites, such as Facebook and MySpace.

Many spam and phishing messages also use this trick. Most users are inundated with so much spam and other unwanted e-mail that they often let their guard down and open e-mails and attachments they shouldn't. These e-mails usually look professional and believable. They often dupe people into disclosing information they should never give in exchange for a gift. These social engineering tricks also occur when a hacker who has already broken into the network sends messages or creates fake Internet pop-up windows. The same tricks have occurred through instant messaging and cell-phone messaging.

In some well-publicized incidents, hackers e-mailed their victims a patch purporting to come from Microsoft or another well-known vendor. Users think it looks like a duck and it quacks like a duck — but it's not Bill! The message is actually from a hacker wanting the user to install the "patch," which installs a Trojan-horse keylogger or creates a backdoor into computers and networks. Hackers use these backdoors to hack into the organization's systems or use the victims' computers (known as *zombies*) as launching pads to attack another system. Even viruses and worms can use social engineering.

For instance, the LoveBug worm told users they had a secret admirer. When the victims opened the e-mail, it was too late. Their computers were infected, and perhaps worse, they didn't have a secret admirer.

The *Nigerian 419* e-mail fraud scheme attempts to access unsuspecting people's bank accounts and money. These social engineers — I mean scamsters — offer to transfer millions of dollars to the victim to repatriate a deceased client's funds to the United States. All the victim must provide is personal bank-account information and a little money up front to cover the transfer expenses. Victims then have their bank accounts emptied. This trap has been around for a while, and it's a shame that people still fall for it.

Many computerized social engineering tactics can be performed anonymously through Internet proxy servers, anonymizers, remailers, and basic SMTP servers that have an open relay. When people fall for requests for confidential personal or corporate information, the sources of these social engineering attacks are often impossible to track.

Social Engineering Countermeasures

You have only a few good lines of defense against social engineering. Even with strong security systems, a naïve or untrained user can let the social engineer into the network. Never underestimate the power of social engineers.

Policies

Specific policies help ward off social engineering in the long term in the following areas:

- Classifying data
- Hiring employees and contractors and setting up user IDs
- Establishing acceptable computer usage
- Removing user IDs and employees, contractors, and consultants who no longer work for the organization
- Setting and resetting passwords
- Responding to security incidents, such as suspicious behavior
- Handling proprietary and confidential information
- Escorting guests

These policies must be enforceable and enforced for everyone within the organization. Keep them up to date, and tell your end users about them.

User awareness and training

The best line of defense against social engineering is training employees to identify and respond to social engineering attacks. User awareness begins with initial training for everyone and follows with security awareness initiatives to keep social engineering defenses fresh in everyone's mind. Align training and awareness with specific security policies — you may also want to have a dedicated security training and awareness policy.

Consider outsourcing security training to a seasoned security trainer. Employees often take training more seriously if it comes from an outsider. Outsourcing security training is worth the investment.

While you approach ongoing user training and awareness in your organization, the following tips can help you combat social engineering in the long term:

- Treat security awareness and training as a business investment.
- Train users on an ongoing basis to keep security fresh in their minds.
- Include information privacy and security tasks and responsibilities in everyone's job descriptions.
- Tailor your training content to your audience whenever possible.
- Create a social engineering awareness program for your business functions and user roles.
- Keep your messages as nontechnical as possible.
- Develop incentive programs for preventing and reporting incidents.
- Lead by example.

Share these tips with your users to help prevent social engineering attacks:

- **Never divulge any information unless you can validate that the people requesting the information need it and are who they say they are.** If a request is made over the telephone, verify the caller's identity and call back.
- **Never click an e-mail link that supposedly loads a page with information that needs updating.** This is especially true for unsolicited e-mails.
- **Be careful when sharing personal information on social networking sites, such as Facebook or LinkedIn.** Also, be on the lookout for people claiming to know you or wanting to be your "friend." Their intentions might be malicious.
- **Escort all guests within a building.**
- **Never open e-mail attachments or other files from strangers.**
- **Never give out passwords.**

A few other general suggestions can ward off social engineering:

- **Never let a stranger connect to one of your network jacks or wireless network — even for a few seconds.** A hacker can place a network analyzer, Trojan-horse program, or other malware directly onto your network.
- **Classify your information assets, both hard copy and electronic.** Train all employees how to handle each asset type.
- **Develop and enforce computer media and document destruction policies** that help ensure data is handled carefully and stays where it should be. A good resource for information on destruction policies is `www.pdaconsulting.com/datadp.htm`.
- **Use cross-shredding paper shredders.** Better still, hire a document-shredding company that specializes in confidential document destruction.

These techniques can reinforce the content of formal training:

- New employee orientation, training lunches, e-mails, and newsletters
- Social engineering survival brochure with tips and FAQs
- Trinkets, such as screen savers, mouse pads, sticky notes, pens, and office posters that bear messages that reinforce security principles

  Appendix A lists my favorite security awareness trinket and tool vendors to improve security awareness and education in your organization.

Chapter 6

Physical Security

In This Chapter

- Understanding the importance of physical security
- Q&A with a well-known physical security expert
- Looking for physical security vulnerabilities
- Implementing countermeasures for physical security attacks

I strongly believe that information security is more dependent on nontechnical policies, procedures, and business processes than on the technical hardware and software solutions that many people and vendors swear by. Physical security — *the protection of physical property* — encompasses both technical and nontechnical components.

Physical security is an often-overlooked but critical aspect of an information security program. Your ability to secure your information depends on your ability to secure your site physically. In this chapter, I cover some common physical security weaknesses, as they relate to computers and information security, that you should look out for in your systems. I also outline free and low-cost countermeasures you can implement to minimize your business's physical vulnerabilities.

I don't recommend breaking and entering, which would be necessary to test certain physical security vulnerabilities *fully*. Instead, approach those areas to see how far you *can* get. Take a fresh look — from an outsider's perspective — at the physical vulnerabilities covered in this chapter. You might discover holes in your physical security infrastructure that you had previously overlooked.

Physical Security Vulnerabilities

Whatever your computer- and network-security technology, practically any hack is possible if an attacker is in your building or computer room. That's why looking for physical security vulnerabilities and fixing them before they're exploited is important.

In small companies, some physical security issues might not be a problem. Many physical security vulnerabilities depend on such factors as:

- Size of the building
- Number of buildings or sites
- Number of employees
- Location and number of building entrance and exit points
- Placement of the data centers and other confidential information

Literally thousands of possible physical security vulnerabilities exist. The bad guys are always on the lookout for them — so you should find these vulnerabilities first. Here are some common physical security vulnerabilities I've found when assessing security:

- No receptionist in a building
- No visitor sign-in or escort required for building access
- Employees trusting visitors because they wear vendor uniforms or say they're there to work on the copier or computers
- No access controls on doors
- Physical security CCTV and data center management systems accessible via the network with the default user ID and password
- Doors propped open
- Publicly accessible computer rooms
- Backup media lying around
- Unsecured computer hardware, especially laptops, and software media
- CDs and DVDs with confidential information in trash cans

When these physical security vulnerabilities are exploited, bad things can happen. Perhaps the biggest problem is that unauthorized people can enter your building. After intruders are in your building, they can wander the halls, log on to computers, rummage through the trash, and steal hard-copy documents, CD-ROMs, and even computers out of offices.

A Q&A on physical security with Jack Wiles

In this Q&A session, Jack Wiles, an information security pioneer with over 30 years of experience, answers several questions on physical security and how a lack of it often leads to information insecurity.

How important do you think physical security is in relation to technical-security issues?

I've been asked that question many times in the past, and from decades of experience with both physical and technical security, I have a standard answer. Without question, many of the most expensive technical-security countermeasures and tools often become worthless when physical security is weak. If I can get my team into your building(s) and walk up to someone's desk and log in as that person, I have bypassed all your technical-security systems. In past security assessments, after my team and I entered a building, we always found that people simply thought that we belonged there — that we were employees. We were always friendly and helpful when we came in contact with real employees. They would often return the kindness by helping us with whatever we asked for.

How were you able to get into most of the buildings when you conducted "red team" penetration tests for companies?

In many cases, we just boldly walked into the building and went up the elevator in multistory buildings. If we were challenged, we always had a story ready. Our typical story was that we thought that this was the HR department, and we were there to apply for a job. If we were stopped at the door and told which building to go to for HR, we simply left and then looked for other entrances to that same building. If we found an outside smoking area at a different door, we attempted *tailgating* and simply walked in behind other employees who were reentering the building after finishing their breaks. Tailgating also worked at most entrances that required card access. In my career as a red-team leader, we were never stopped and questioned. We simply said "thank you" as we walked in and compromised the entire building.

What kinds of things would you bring out of a building?

It was always easy to get enough important documentation to prove that we were there. In many cases, the documentation was sitting in a recycle box next to someone's desk (especially if that person was someone important). To us, that really said, "Steal me first!" We found it interesting that many companies just let their recycle boxes fill up before emptying them. We would also look for a room where strip-cut shredders were used. The documents that were shredded were usually stored in clear plastic bags. We loaded these bags into our cars and had many of the shredded documents put back together in a few hours. We found that if we pasted the strips from any page on cardboard with as much as an inch of space between the strips, the final document was still readable.

Jack Wiles is president of TheTrainingCo. (`www.thetrainingco.com`) and promotes the annual information security conference Techno Security.

What to Look For

You should look for some specific security vulnerabilities. Many potential physical security exploits seem unlikely, but they can occur to organizations that don't take physical security seriously.

The bad guys can exploit many physical security vulnerabilities, including weaknesses in a building's infrastructure, office layout, computer-room access, and design. In addition to these factors, consider the facility's proximity to local emergency assistance (police, fire, and ambulance) and the area's crime statistics (burglary, breaking and entering, and so on) so you can better understand what you're up against.

Look for the vulnerabilities discussed in the following sections when assessing your organization's physical security. This won't take a lot of technical savvy or expensive equipment. Depending on the size of your facilities, these tests shouldn't take much time either. The bottom line is to determine whether the physical security controls are adequate for the risks involved. Above all, be practical and use common sense.

Building infrastructure

Doors, windows, and walls are critical components of a building — especially for a computer room or area where confidential information is stored.

Attack points

Hackers can exploit a handful of building infrastructure vulnerabilities. Consider the following commonly overlooked attack points:

- Are doors propped open? If so, why?
- Can gaps at the bottom of critical doors allow someone using a balloon or other device to trip a sensor on the inside of a "secure" room?
- Would it be easy to force doors open? Would a simple kick near the doorknob suffice?
- What is the building or computer room made of (steel, wood, concrete), and how sturdy are the walls and entryways? How resilient is the material to earthquakes, tornadoes, strong winds, heavy rains, and vehicles driving into the building — would these disasters leave the building exposed so that looters and others with malicious intent could gain access to the computer room or other critical areas?
- Are any doors or windows made of glass? Are hinges on the outside? Is this glass clear? Is the glass shatterproof or bulletproof?
- Are doors, windows, and other entry points wired to an alarm system?

- Are there *drop ceilings* with tiles that can be pushed up? Are the walls slab-to-slab? If not, someone could easily scale walls, bypassing any door or window access controls.

Countermeasures

Many physical security countermeasures for building vulnerabilities might require other maintenance, construction, or operations experts. If building infrastructures is not your forte, you can hire outside experts during the design, assessment, and retrofitting stages to ensure that you have adequate controls. Here are some of the best ways to solidify building security:

- Strong doors and locks
- Windowless walls around computer rooms
- A continuously monitored alarm system with network-based cameras located at all access points
- Lighting (especially around entry and exit points)
- Mantraps and sallyports that allow only one person at a time to pass through a door
- Fences (with barbed wire or razor wire)

Utilities

You must consider building and computer room utilities, such as power, water, and fire suppression, when assessing physical security. These utilities can help fight off such incidents as fire and keep other access controls running during a power loss. They can also be used against you if an intruder enters the building.

Attack points

Intruders often exploit utility-related vulnerabilities. Consider the following attack points, which are commonly overlooked:

- Is power-protection equipment (surge protectors, UPSes, and generators) in place? How easily accessible are the on/off switches on these devices? Can an intruder walk in and flip a switch?
- When the power fails, what happens to physical security mechanisms? Do they fail *open,* allowing anyone through, or fail *closed,* keeping everyone in or out until the power is restored?
- Where are fire detection and suppression devices — including alarm sensors, extinguishers, and sprinkler systems — located? Determine how a malicious intruder can abuse them. Are these devices placed where they can harm electronic equipment during a false alarm?

- Where are water and gas shutoff valves located? Can you access them, or would you have to call maintenance personnel about an incident?
- Are local telecom wires (both copper and fiber) that run outside of the building located aboveground, where someone can tap into them with telecom tools? Can digging in the area cut them easily? Are they located on telephone poles that are vulnerable to traffic accidents?

Countermeasures

You might need to involve other experts during the design, assessment, or retrofitting stages. The key is *placement:*

- Where are the major utility controls placed? Ideally, they need to be behind closed and lockable doors out of sight to people passing through.
- Can someone walking through the building access the controls to turn them on and off?

Covers for on/off switches and thermostat controls and locks for server power buttons, USB ports, and PCI expansion slots are effective defenses.

I once assessed the physical security of an Internet colocation facility for a very large computer company (that will remain anonymous). I made it past the front guard and tailgated through all the controlled doors to reach the data center. After I was inside, I walked by equipment that was owned by very large dot-com companies, such as servers, routers, firewalls, UPSes, and power cords. All this equipment was completely exposed to anyone walking in that area. A quick flip of a switch or an accidental trip over a network cable dangling to the floor could bring an entire shelf — and a global e-commerce site — to the ground.

Office layout and usage

Office design and usage can either help or hinder physical security.

Attack points

Hackers might exploit some office vulnerabilities. Consider these attack points:

- Does a receptionist or security guard monitor traffic in and out of the main doors of the building?
- Do employees have confidential information on their desks? What about mail and other packages — do they lie around outside someone's door or, even worse, outside the building, waiting for pickup?

- Where are trash cans and dumpsters located? Are they easily accessible by anyone? Are recycling bins or shredders used?

  Open recycling bins and other careless handling of trash are open invitations for dumpster diving, when hackers search for confidential company information, such as phone lists and memos, in the trash. Dumpster diving can lead to many security exposures.

- How secure are the mail and copy rooms?

  If hackers can access these rooms, they can steal mail or company letterhead to use against you. They can also use and abuse your fax machine(s).

- Are closed-circuit television (CCTV) cameras used *and* monitored?

- What access controls are on doors and windows? Are regular keys, card keys, combination locks, or biometrics used? Who can access these keys, and where are they stored?

  Keys and programmable keypad combinations are often shared among users, making accountability difficult to determine. Find out how many people share these combinations and keys.

I came across a situation for a client where the front lobby entrance was unmonitored. It also happened to have a Voice over IP (VoIP) phone available for anyone to use. But they did not consider that anyone could enter the lobby, disconnect the VoIP phone, and plug a laptop computer into the connection and have full access to their network with minimal chance that the intruder would ever be questioned about what he or she was doing. This could have been prevented had the voice and data traffic been separated.

Countermeasures

Putting simple measures, such as the following, in place can reduce your exposure to office vulnerabilities:

- A receptionist or a security guard who monitors people coming and going. This is the most critical countermeasure. This person can ensure that every visitor signs in and that all new or untrusted visitors are always escorted.

  Make it policy and procedure for all employees to question strangers and report strange behavior in the building.

  Employees Only or *Authorized Personnel Only* signs show the bad guys where they *should* go instead of deterring them from entering.

- CCTV cameras.

- Single entry and exit points to a data center.

- Secure areas for dumpsters.

- Cross-cut shredders or secure recycling bins for hard-copy documents.
- Limited numbers of keys and passcode combinations.

  Make keys and passcodes unique for each person whenever possible.
- Biometrics identification systems can be very effective, but they can also be expensive and difficult to manage.

Network components and computers

After hackers obtain physical access to a building, they look for the computer room and other easily accessible computer and network devices.

Attack points

The keys to the kingdom are often as close as someone's desktop computer and not much farther than an unsecured computer room or wiring closet.

Malicious intruders can do the following:

- Obtain network access and send malicious e-mails as a logged-in user.
- Crack and obtain passwords directly from the computer by booting it with a tool such as the Ophcrack Live CD (`http://ophcrack.sourceforge.net`). I cover this tool and more password hacks in Chapter 7.
- Steal files from the computer by copying them to a removable storage device such as a smartphone, MP3 player, or USB drive or e-mailing them to an external address.
- Enter unlocked computer rooms and mess around with servers, firewalls, and routers.
- Walk out with network diagrams, contact lists, and business-continuity and incident-response plans.
- Obtain phone numbers from analog lines and circuit IDs from T1, frame-relay, and other telecom equipment for future attacks.

Practically every bit of unencrypted information that traverses the network can be recorded for future analysis through one of the following methods:

- Connecting a computer running network-analyzer software to a hub or monitor, or a mirrored port on a switch on your network.
- Installing network analyzer software on an existing computer.

  This is very hard to spot.

How would hackers access this information in the future?

- The easiest attack method is to install remote-administration software on the computer, such as VNC (`www.realvnc.com`).
- A crafty hacker with enough time can bind a public IP address to the computer if the computer is outside the firewall. Hackers with enough network knowledge can configure new firewall rules to do this.

Also, consider these other vulnerabilities:

- How easily can someone's computer be accessed during regular business hours? During lunchtime? After hours?
- Are servers, firewalls, routers, and switches mounted in locked racks?
- Are computers — especially laptops — secured to desks with locks? Are their hard drives encrypted in the event one is lost or stolen?
- Are passwords stored on sticky notes on computer screens, keyboards, or desks?
- Are backup media lying around the computer room susceptible to theft?
- Are safes used to protect backup media? Are they specifically rated for media to keep backups from melting during a fire? Who can access the safe?
- How are laptops and hand-held computers handled in-house and when employees are working from home or traveling? Are smartphones sitting around unsecured?

  These devices are often at great risk because of their size and value. Also, they are typically unprotected by the organization's regular security controls. Are specific policies and technologies in place to help protect them? Are locking laptop bags required? What about power-on passwords? Also, consider encryption in case these devices get into a hacker's hands.
- How easily can someone access a wireless access point (AP) signal or the AP itself to join the network? Rogue access points are also something to consider.
- Are network firewalls, routers, switches, and hubs (basically, anything with an Ethernet connection) easily accessible, which would enable a hacker to plug in to the network easily?
- Are all cables patched through on the patch panel in the wiring closet so all network drops are live?

  This set-up is very common but a bad idea because it allows anyone to plug in to the network anywhere and gain access.
- Are cable traps and locks that prevent hackers from unplugging network cables from patch panels or computers and using those connections for their computers in place?

Countermeasures

Network and computer security countermeasures are some of the simplest to implement yet the most difficult to enforce because they involve everyday actions. Here is a rundown of these countermeasures:

- **Require users to lock their screens —** which usually takes a few clicks or keystrokes in Windows or UNIX — when they leave their computers.
- **Ensure that strong passwords are used.** I cover this in Chapter 7.
- **Require laptop users to lock their systems to their desks with a locking cable.** This is especially important in larger companies or locations that receive a lot of foot traffic.
- **Require all laptops to use whole disk encryption technologies,** such as Windows BitLocker (`www.microsoft.com/windows/windows-vista/features/bitlocker.aspx`)and PGP's Whole Disk Encryption product (`www.pgp.com/products/wholediskencryption/index.html`).
- **Keep computer rooms and wiring closets locked, and monitor those areas for wrongdoings.**
- **Keep a current inventory of hardware and software within the organization so it's easy to determine when extra equipment appears or equipment is missing.** This is especially important in computer rooms.
- **Properly secure computer media when stored and during transport.**
- **Scan for rogue wireless access points.** I discuss wireless networks in depth in Chapter 9.
- **Use a bulk eraser on magnetic media before they're discarded.**

Chapter 7

Passwords

In This Chapter

- Identifying password vulnerabilities
- Examining password-hacking tools and techniques
- Hacking operating system passwords
- Hacking password-protected files
- Protecting your systems from password hacking

Password hacking is one of the easiest and most common ways attackers obtain unauthorized computer or network access. Although strong passwords — ideally, longer and stronger passphrases that are difficult to *crack* (or guess) — are easy to create and maintain, network administrators and users often neglect this. Therefore, passwords are one of the weakest links in the information security chain. Passwords rely on secrecy. After a password is compromised, its original owner isn't the only person who can access the system with it. That's when accountability goes out the window and bad things start happening.

External attackers and malicious insiders have many ways to obtain passwords. They can glean passwords simply by asking for them or by looking over the shoulders *(shoulder surfing)* of users while they type them. Hackers can also obtain passwords from local computers by using password-cracking software. To obtain passwords from across a network, attackers can use remote cracking utilities, keyloggers, or network analyzers.

This chapter demonstrates how easily the bad guys can gather password information from your network and computer systems. I outline common password vulnerabilities and describe countermeasures to help prevent these vulnerabilities from being exploited on your systems. If you perform the tests and implement the countermeasures outlined in this chapter, you'll be well on your way to securing your systems' passwords.

Password Vulnerabilities

When you balance the cost of security and the value of the protected information, the combination of a *user ID* and a *secret password* is usually adequate. However, passwords give a false sense of security. The bad guys know this and attempt to crack passwords as a step toward breaking into computer systems.

One big problem with relying solely on passwords for information security is that more than one person can know them. Sometimes, this is intentional; often, it's not. The tough part is that there's no way of knowing who, besides the password's owner, knows a password. ***Remember:*** Knowing a password doesn't make someone an authorized user.

Here are the two general classifications of password vulnerabilities:

- **Organizational or user vulnerabilities:** This includes lack of password policies that are enforced within the organization and lack of security awareness on the part of users.
- **Technical vulnerabilities:** This includes weak encryption methods and unsecure storage of passwords on computer systems.

Before computer networks and the Internet, the user's physical environment was an additional layer of password security that actually worked pretty well. Now that most computers have network connectivity, that protection is gone.

Organizational password vulnerabilities

It's human nature to want convenience — especially when it comes to remembering five, ten, and often dozens of passwords in our work and daily lives. This makes passwords one of the easiest barriers for an attacker to overcome. Almost 3 trillion (yes, trillion with a *t* and 12 zeros) eight-character password combinations are possible by using the 26 letters of the alphabet and the numerals 0 through 9. However, most people prefer to create passwords that are easy to remember. Users like to use such passwords as *password*, their login name, or even a blank password.

A case study in Windows password vulnerabilities with Philippe Oechslin

In this case study, Dr. Philippe Oechslin, a researcher and independent information security consultant, shared with me his recent research findings on Windows password vulnerabilities.

The Situation

In 2003, Dr. Oechslin discovered a new method for cracking Windows passwords — now commonly referred to as *rainbow cracking*. While testing a brute-force password-cracking tool, Dr. Oechslin thought that everyone using the same tool to generate the same hashes repeatedly was a waste of time. He believed that generating a huge dictionary of all possible hashes would make it easier to crack Windows passwords, but then quickly realized that a dictionary of the LAN Manager (LM) hashes of all possible alphanumerical passwords would require over a terabyte of storage.

During his research, Dr. Oechslin discovered a technique called *time-memory trade-offs,* where hashes are computed in advance, but only a small fraction are stored (approximately one in a thousand). Dr. Oechslin discovered that how the LM hashes are organized allows you to find any password if you spend some time recalculating some of the hashes. This technique saves memory but takes a lot of time. Studying this method, Dr. Oechslin found a way to make the process more efficient, making it possible to find any of the 80 billion unique hashes by using a table of 250 million entries (1GB worth of data) and performing only 4 million hash calculations. This process is much faster than a brute-force attack, which must generate 50 percent of the hashes (40 billion) on average.

This research is based on the absence of a random element when Windows passwords are hashed. This is true for both the LM hash and the NTLM hash built in to Windows. As a result, the same password produces the same hash on any Windows machine. Although it is known that Windows hashes have no random element, no one has used a technique like the one that Dr. Oechslin discovered to crack Windows passwords.

Dr. Oechslin and his team originally placed an interactive tool on their Web site (`http://lasecwww.epfl.ch`) that enabled visitors to submit hashes and have them cracked. Over a six-day period, the tool cracked 1,845 passwords in an average of 7.7 seconds! You can try out the demo for yourself at `www.objectif-securite.ch/en/products.php`.

The Outcome

So what's the big deal, you say? This password-cracking method can crack practically any alphanumeric password in a few seconds, whereas current brute-force tools can take several hours. Dr. Oechslin and his research team have generated a table with which they can crack any password made of letters, numbers, and 16 other characters in less than a minute, demonstrating that passwords made up of letters and numbers aren't good enough. Philippe also stated that this method is useful for ethical hackers who have only limited time to perform their testing. Unfortunately, malicious hackers have the same benefit and can perform their attacks before anyone detects them!

Philippe Oechslin, PhD, CISSP, is a lecturer and senior research assistant at the Swiss Federal Institute of Technology in Lausanne and is founder and CEO of Objectif Sécurité (`www.objectif-securite.ch/en`).

Unless users are educated and reminded about using strong passwords, their passwords usually are

- **Easy to guess.**
- **Seldom changed.**
- **Reused for many security points.** When bad guys crack one password, they can often access other systems with that same password and username.
- **Written down in unsecure places.** The more complex a password is, the more difficult it is to crack. However, when users create complex passwords, they're more likely to write them down. External attackers and malicious insiders can find these passwords and use them against you.

Technical password vulnerabilities

You can often find these serious technical vulnerabilities after exploiting organizational password vulnerabilities:

- **Weak password encryption schemes.** Hackers can break weak password storage mechanisms by using cracking methods that I outline in this chapter. Many vendors and developers believe that passwords are safe as long as they don't publish the source code for their encryption algorithms. *Wrong!* A persistent, patient attacker can usually crack this *security by obscurity* (a security measure that's hidden from plain view but can be easily overcome) fairly quickly. After the code is cracked, it is distributed across the Internet and becomes public knowledge.

  Password-cracking utilities take advantage of weak password encryption. These utilities do the grunt work and can crack any password, given enough time and computing power.
- **Programs that store their passwords in memory, unsecured files, and easily accessed databases.**
- **User applications that display passwords on the screen while typing.**

The National Vulnerability Database (an index of computer vulnerabilities managed by the National Institute of Standards and Technology) currently identifies over 2,000 password-related vulnerabilities — a number that has doubled in just the past three years! You can search for these issues at (`http://nvd.nist.gov`) to find out how vulnerable some of your systems are from a technical perspective.

Cracking Passwords

Password cracking is one of the most enjoyable hacks for the bad guys. It fuels their sense of exploration and desire to figure out things. You might not have a burning desire to explore everyone's passwords, but it helps to approach password cracking with this mindset. So where should you start hacking the passwords on your systems? Generally, any user's password works. After you obtain one password, you can often obtain others — including administrator or root passwords.

Administrator passwords are the pot of gold. With unauthorized administrative access, you can do virtually anything on the system. When looking for your organization's password vulnerabilities, I recommend first trying to obtain the highest level of access possible (such as administrator) through the most discreet method possible. That's often what the bad guys do.

You can use low-tech ways and high-tech ways to exploit vulnerabilities to obtain passwords. For example, you can deceive users into divulging passwords over the telephone or simply observe what a user has written down on a piece of paper. Or you can capture passwords directly from a computer, over a network, and via the Internet with the tools covered in the following sections.

Cracking passwords the old-fashioned way

A hacker can use low-tech methods to crack passwords. These methods include using social engineering techniques, shoulder surfing, and simply guessing passwords from information that he knows about the user.

Social engineering

The most popular low-tech method for gathering passwords is *social engineering,* which I cover in detail in Chapter 5. Social engineering takes advantage of the trusting nature of human beings to gain information that later can be used maliciously. A common social engineering technique is simply to con people into divulging their passwords. It sounds ridiculous, but it happens all the time.

Techniques

To obtain a password through social engineering, you just ask for it. For example, you can simply call a user and tell him that he has some important-looking e-mails stuck in the mail queue, and you need his password to log in and free them up. This is often how hackers and rogue insiders try to get the information!

If a user gives you his password during your testing, make sure that he changes it. You don't want to be held accountable if something goes awry after the password has been disclosed.

Countermeasures

User awareness and consistent security training is the best defense against social engineering. Train users to spot attacks (such as suspicious phone calls or deceitful phishing e-mails) and respond effectively. Their best response is not to give out any information and to alert the appropriate information security manager in the organization to see whether the inquiry is legitimate and whether a response is necessary.

Shoulder surfing

Shoulder surfing (the act of looking over someone's shoulder to see what the person is typing) is an effective, low-tech password hack.

Techniques

To mount this attack, the bad guys must be near their victims and not look obvious. They simply collect the password by watching either the user's keyboard or screen when the person logs in. An attacker with a good eye might even watch whether the user is glancing around his desk for either a reminder of the password or the password itself.

You can try shoulder surfing yourself. Simply walk around the office and perform random spot checks. Go to users' desks and ask them to log in to their computers, the network, or even their e-mail applications. Just don't tell them what you're doing beforehand, or they might attempt to hide what they're typing or where they're looking for their password — two things that they should've been doing all along! Just be careful doing this and respect other people's privacy.

Countermeasures

Encourage users to be aware of their surroundings and not to enter their passwords when they suspect that someone is looking over their shoulders. Instruct users that if they suspect someone is looking over their shoulders while they're logging in, they should politely ask the person to look away or, do what I do often, just lean into their line of sight to keep them from seeing my typing and/or computer screen. 3M Privacy Filters (`www.3m.com`) work great as well.

Inference

Inference is simply guessing passwords from information you know about users — such as their date of birth, favorite television show, or phone numbers. It sounds silly, but criminals often determine their victims' passwords simply by guessing them!

The best defense against an inference attack is to educate users about creating secure passwords that do not include information that can be associated with them. Outside of certain password complexity filters, it's often not easy to enforce this practice with technical controls, so you need a sound security policy and ongoing security awareness and training to remind users of the importance of secure password creation.

Weak authentication

External attackers and malicious insiders can obtain — or simply avoid having to use — passwords by taking advantage of older operating systems, such as Windows 9*x* and Windows ME. These operating systems don't require passwords to log in. The same goes for a BlackBerry or smartphone that isn't configured to use passwords.

Bypassing authentication

On a Windows 9*x* or similar workstation that prompts for a password, you can press Esc on the keyboard to get right in. After you're in, you can find other passwords stored in such places as dial-up and VPN connections and screen savers. Such passwords can be cracked very easily using Elcomsoft's Proactive System Password Recovery tool (`www.elcomsoft.com/pspr.html`) and Cain & Abel (`www.oxid.it/cain.html`). These weak systems can serve as *trusted* machines — meaning that people assume they're secure — and provide good launching pads for network-based password attacks as well.

Countermeasures

The only true defense against this is not to use operating systems that employ weak authentication. To eliminate this vulnerability, *at least* upgrade to Windows XP, or better yet, Windows 7 or use recent versions of Linux or one of the various flavors of UNIX, including Mac OS X.

More modern authentication systems, such as Kerberos (which is used in newer versions of Windows) and directory services (such as Novell's eDirectory and Microsoft's Active Directory), encrypt user passwords or don't communicate the passwords across the network at all, which creates an extra layer of security.

High-tech password cracking

High-tech password cracking involves using a program that tries to guess a password by determining all possible password combinations. These high-tech methods are mostly automated after you access the computer and password database files.

The main password-cracking methods are dictionary attacks, brute-force attacks, and rainbow attacks.

Password-cracking software

You can try to crack your organization's operating system and application passwords with various password-cracking tools:

- **Cain & Abel** (`www.oxid.it/cain.html`) cracks LM and NT LanManager (NTLM) hashes, Windows RDP passwords, Cisco IOS and PIX hashes, VNC passwords, RADIUS hashes, and lots more.
- **chknull** (`www.phreak.org/archives/exploits/novell`) checks for Novell NetWare accounts with no password.
- **Elcomsoft Distributed Password Recovery** (`www.elcomsoft.com/edpr.html`) cracks Microsoft Office, PGP, and PKCS passwords in a distributed fashion using up to 10,000 networked computers at one time. Plus, this tool uses the same GPU video acceleration as the Elcomsoft Wireless Auditor tool, which allows for cracking speeds up to 50 times faster. (I talk about the Elcomsoft Wireless Auditor tool in Chapter 9.)
- **Elcomsoft System Recovery** (`www.elcomsoft.com/esr.html`) cracks or resets Windows user passwords, sets administrative rights, and resets password expirations all from a bootable CD.
- **John the Ripper** (`www.openwall.com/john`) cracks hashed Linux/UNIX and Windows passwords.
- **ophcrack** (`http://ophcrack.sourceforge.net`) cracks Windows user passwords using rainbow tables from a bootable CD.
- **Pandora** (`www.nmrc.org/project/pandora`) cracks Novell NetWare passwords online and offline.
- **Proactive Password Auditor** (`www.elcomsoft.com/ppa.html`) runs brute-force, dictionary, and rainbow cracks against extracted LM and NTLM password hashes.
- **Proactive System Password Recovery** (`www.elcomsoft.com/pspr.html`) recovers practically any locally stored Windows password, such as logon passwords, WEP/WPA passphrases, SYSKEY passwords, and RAS/dialup/VPN passwords.
- **pwdump3** (`www.openwall.com/passwords/dl/pwdump/pwdump3v2.zip`) extracts Windows password hashes from the SAM database.
- **RainbowCrack** (`http://project-rainbowcrack.com`) cracks LanManager (LM) and MD5 hashes very quickly by using rainbow tables.

Some of these tools require physical access to the systems you're testing. You might be wondering what value that adds to password cracking. If a hacker can obtain physical access to your systems and password files, you have more than just basic information security problems to worry about, right? True, but this kind of access is entirely possible! What about a summer intern, a disgruntled employee, or an outside auditor with malicious intent?

Password-cracking utilities take a set of known passwords and run them through a password-hashing algorithm. The resulting encrypted hashes are then compared at lightning speed to the password hashes extracted from the original password database. When a match is found between the newly generated hash and the hash in the original database, the password has been cracked. It's that simple.

Other password cracking programs simply attempt to log on using a predefined set of user IDs and passwords. This is how many dictionary-based cracking tools work, such as Brutus (`www.hoobie.net/brutus`) and SQLPing3 (`www.sqlsecurity.com/Tools/FreeTools/tabid/65/Default.aspx`). I cover cracking Web application and database passwords in Chapters 14 and 15.

Passwords that are subjected to cracking tools eventually lose. You have access to the same tools as the bad guys. These tools can be used for both legitimate security assessments and malicious attacks. You want to find password weaknesses before the bad guys do, and in this section, I show you some of my favorite methods for assessing Windows and Linux/UNIX passwords.

When trying to crack passwords, the associated user accounts might be locked out, which could interrupt your users. Be careful if your systems intruder lockout is enabled — otherwise, you might lock out some or all computer/network accounts, resulting in a sort of denial of service situation for your users.

Passwords are typically encrypted when they're stored on a computer, using an encryption or one-way hash algorithm, such as DES or MD5. Hashed passwords are then represented as fixed-length encrypted strings that always represent the same passwords with exactly the same strings. These hashes are irreversible for all practical purposes, so, in theory, passwords can never be decrypted. Furthermore, certain passwords, such as those in Linux, have a random value called a "salt" added to them to create a degree of randomness. This prevents the same password used by two people from having the same hash value.

Password storage locations vary by operating system:

- Windows usually stores passwords in these locations:
  - Security Accounts Manager (SAM) database (`c:\winnt\system32\config`)
  - Active Directory database file that's stored locally or spread across domain controllers (`ntds.dit`)

  Windows sometimes stores passwords in either a backup of the SAM file in the `c:\winnt\repair` directory or on an emergency repair disk.

Some Windows applications store passwords in the Registry or as plain-text files on the hard drive!

- Linux and other UNIX variants typically store passwords in these files:
  - `/etc/passwd` (readable by everyone)
  - `/etc/shadow` (accessible by the system and the root account only)
  - `/etc/security/passwd` (accessible by the system and the root account only)
  - `/.secure/etc/passwd` (accessible by the system and the root account only)

Dictionary attacks

Dictionary attacks quickly compare a set of known dictionary-type words — including many common passwords — against a password database. This database is a text file with hundreds if not thousands of "dictionary" words typically listed in alphabetical order. For instance, suppose that you have a dictionary file that you downloaded from one of the sites in the following list. The English dictionary file at the Purdue site contains one word per line starting with *10th, 1st* . . . all the way to *zygote.*

Many password-cracking utilities can use a separate dictionary that you create or download from the Internet. Here are some popular sites that house dictionary files and other miscellaneous word lists:

- `ftp://ftp.cerias.purdue.edu/pub/dict`
- `ftp://ftp.ox.ac.uk/pub/wordlists`
- `http://packetstormsecurity.nl/Crackers/wordlists`
- `www.outpost9.com/files/WordLists.html`

The above links are good but I find the BlackKnightList (`http://rs159.rapidshare.com/files/184075601/BlackKnightList.rar`) is the most comprehensive. After you download the file, you need WinRAR (`www.rarlab.com`) or a similar program to open it and gain access to the text file inside.

Don't forget to use other language files as well, such as Spanish and Klingon.

Dictionary attacks are only as good as the dictionary files you supply your password-cracking program.

Most dictionary attacks are good for *weak* (easily guessed) passwords. However, some special dictionaries have common misspellings or alternate spellings of words, such as `pa$$w0rd` (password) and `5ecur1ty` (security).

Additionally, special dictionaries can contain non-English words and thematic words from religions, politics, or *Star Trek*.

Brute-force attacks

Brute-force attacks can crack practically any password, given sufficient time. Brute-force attacks try every combination of numbers, letters, and special characters until the password is discovered. Many password-cracking utilities let you specify such testing criteria as the character sets, password length to try, and known characters (for a "mask" attack). Sample Proactive Password Auditor brute-force password-cracking options are shown in Figure 7-1.

A brute-force test can take quite a while, depending on the number of accounts, their associated password complexities, and the speed of the computer that's running the cracking software. As powerful as brute-force testing can be, it literally can take forever to exhaust all possible password combinations, which in reality, is not practical in every situation.

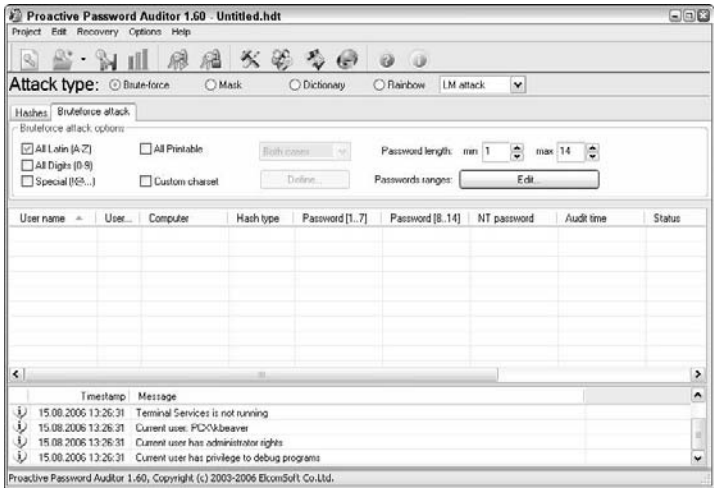


Figure 7-1: Brute-force password-cracking options in Proactive Password Auditor.

Smart hackers attempt logins slowly or at random times so the failed login attempts aren't as predictable or obvious in the system log files. Some malicious users might even call the IT help desk to attempt a reset of the account they just locked out. This social-engineering technique could be a major issue, especially if the organization has no (or minimal) mechanisms in place to verify that locked-out users are who they say they are.

Can an expiring password deter a hacker's attack and render password-cracking software useless? Yes. After the password is changed, the cracking must start again if the hacker wants to test all the possible combinations.

This is one reason why it's often a good idea to change passwords periodically. Shortening the change interval can reduce the risk of passwords being cracked. Refer to the U.S. Department of Defense's Password Management Guideline document (`www.itl.nist.gov/fipspubs/app-e.htm`) for more information on this topic.

Exhaustive password cracking attempts usually aren't necessary. Most passwords are fairly weak. Even minimum password requirements, such as a password length, can help you in your testing; you might be able to discover security policy information by using other tools (see Part IV for tools and techniques for testing the security of operating systems) and configure your cracking programs with more well-defined cracking parameters, which often generate faster results.

Rainbow attacks

A rainbow password attack uses rainbow tables (see the earlier sidebar, "A case study in Windows password vulnerabilities with Philippe Oechslin") to crack various password hashes for LM, NTLM, Cisco PIX, and MD5 much more quickly and with extremely high success rates (near 100%). Password-cracking speed is increased in a rainbow attack because the hashes are precalculated, thus, don't have to be generated individually on the fly as they are with dictionary and brute-force cracking methods.

Unlike dictionary and brute-force attacks, rainbow attacks cannot be used to crack password hashes of unlimited length. The current maximum length for Microsoft LM hashes is 14 characters and the tables are available for purchase and download via the ophcrack site at `http://ophcrack.sourceforge.net`. There's a length limitation because it takes *significant* time to generate these rainbow tables. Given enough time, a sufficient number of tables will be created. Of course, by then, computers and applications likely have different authentication mechanisms and hashing standards — including a new set of vulnerabilities — to contend with.

If you have a good set of rainbow tables, such as those offered via the ophcrack site and Project RainbowCrack (`http://project-rainbowcrack.com`), you can crack passwords in seconds, minutes, or hours versus the days, weeks, or even years required by dictionary and brute-force methods.

Cracking Windows passwords with pwdump3 and John the Ripper

The following steps use two of my favorite utilities to test the security of current passwords on Windows systems:

- pwdump3 (to extract password hashes from the Windows SAM database)
- John the Ripper (to crack the hashes of Windows and Linux/UNIX passwords)

This test requires administrative access to either your Windows standalone workstation or the server:

1. **Create a new directory called `passwords` from the root of your Windows C: drive.**
2. **Download and install a decompression tool if you don't already have one.**

WinZip (`www.winzip.com`) is a good commercial tool I use and FreeZip (`http://members.ozemail.com.au/~nulifetv/freezip`) is a free decompression tool. Windows XP, Windows Vista, and Windows 7 also include built-in zip file handling.

3. **Download, extract, and install the following software into the `passwords` directory you created, if you don't already have it on your system:**
   - *pwdump3:* Download the file from `www.openwall.com/passwords/dl/pwdump/pwdump3v2.zip`
   - *John the Ripper:* Download the file from `www.openwall.com/john`
4. **Enter the following command to run pwdump3 and redirect its output to a file called `cracked.txt`:**

```
c:\passwords\pwdump3 > cracked.txt
```

This file capture the Windows SAM password hashes that are cracked with John the Ripper. Figure 7-2 shows the contents of the `cracked.txt` file that contains the local Windows SAM database password hashes.

Figure 7-2: Output from pwdump3.

5. **Enter the following command to run John the Ripper against the Windows SAM password hashes to display the cracked passwords:**

```
c:\passwords\john cracked.txt
```

This process — shown in Figure 7-3 — can take seconds or days, depending on the number of users and the complexity of their associated passwords. My Windows example took only five seconds to crack five weak passwords.

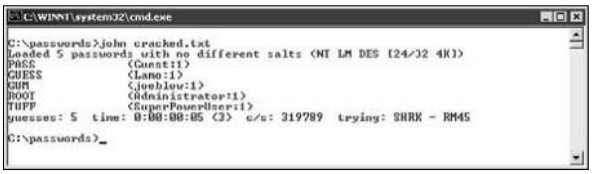


Figure 7-3: Cracked password file hashes using John the Ripper.

Cracking UNIX passwords with John the Ripper

John the Ripper can also crack UNIX passwords. You need root access to your system and to the password (`/etc/passwd`) and shadow password (`/etc/shadow`) files. Perform the following steps for cracking UNIX passwords:

1. **Download the UNIX source files from `www.openwall.com/john`.**
2. **Extract the program by entering the following command:**

   ```
   [root@localhost kbeaver]#tar -zxf john-1.7.1.tar.gz
   ```


   You can crack UNIX passwords on a Windows system using the Windows/DOS version of John the Ripper.
3. **Change to the `/src` directory that was created when you extracted the program and enter the following command:**

   ```
   make generic
   ```
4. **Change to the `/run` directory and enter the following command to use the unshadow program to combine the `passwd` and `shadow` files and copy them to the file `cracked.txt`:**

   ```
   ./unshadow /etc/passwd /etc/shadow > cracked.txt
   ```


   This will not work with all UNIX variants.
5. **Enter the following command to start the cracking process:**

   ```
   ./john cracked.txt
   ```

   When John the Ripper is complete (and this could take some time), the output is similar to the results of the preceding Windows process. (Refer to Figure 7-3.)

After completing the preceding Windows or UNIX steps, you can either force users to change passwords that don't meet specific password policy requirements, or create a new password policy.

Be careful handling the results of your password cracking. You create an accountability issue because more than one person now knows the passwords. Always treat the password information of others as strictly confidential.

Passwords by the numbers

One hundred twenty-eight different ASCII characters are used in typical computer passwords. (Technically, only 126 characters are used because you can't use the NULL and the carriage return characters.) A truly random eight-character password that uses 126 different characters can have 63,527,879,748,485,376 different combinations. Taking that a step further, if it were possible (and it is in Linux and UNIX) to use all 256 ASCII characters (254, without NULL and carriage return) in a password, 17,324,859,965,700,833,536 different combinations would be available. This is approximately 2.7 billion times more combinations than there are people on earth!

A text file containing all the possible passwords would require millions of terabytes of storage space. Even if you include only the more realistic combination of 95 or so ASCII letters, numbers, and standard punctuation characters, such a file would still fill thousands of terabytes of storage space. These storage requirements require dictionary and brute-force password-cracking programs to form the password combinations on the fly, instead of reading all possible combinations from a text file. That's why rainbow attacks are more effective at cracking passwords than dictionary and brute-force attacks.

Given the effectiveness of rainbow password attacks, it's realistic to think that eventually, anyone will be able to crack all possible password combinations, given the current technology and average lifespan. It probably won't happen; however, many thought in the 1980s that 640KB of RAM and a 10MB hard drive in a PC were all that would ever be needed!

Cracking Windows passwords using rainbow tables with ophcrack

You can also perform a rainbow attack by using the open source tool ophcrack (not to be confused with the retired L0phtcrack). Perform the following steps for the Windows version:

1. **Download the source file from `http://ophcrack.sourceforge.net`.**
2. **Extract and install the program by entering the following command:**

   `ophcrack-win32-installer-3.3.1.exe` (or whatever the current filename is)
3. **Load the program.**
4. **Click the Load button and select the type of test you wish to run.**

   In this example, shown in Figure 7-4, I'm connecting to a remote server called `test1`. This way, ophcrack will authenticate to the remote server using my locally logged-in username and run pwdump code to extract the password hashes from the server's SAM database. You can also load hashes from the local machine or from hashes extracted during a previous pwdump session.

   The extracted password hash usernames will look similar to those shown in Figure 7-5.

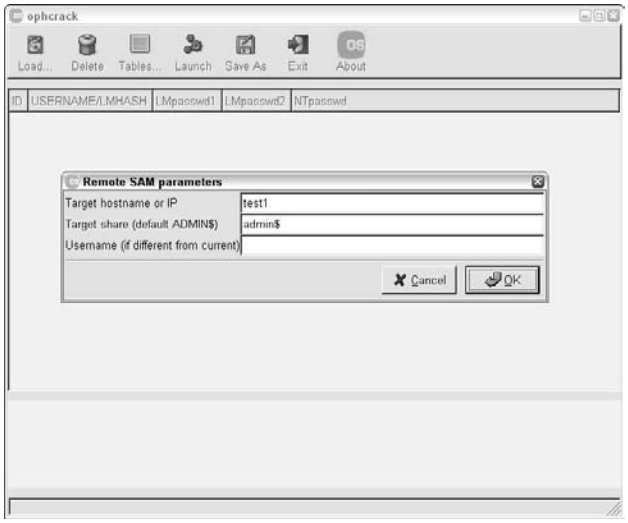


Figure 7-4: Loading password hashes from a remote SAM database in ophcrack.

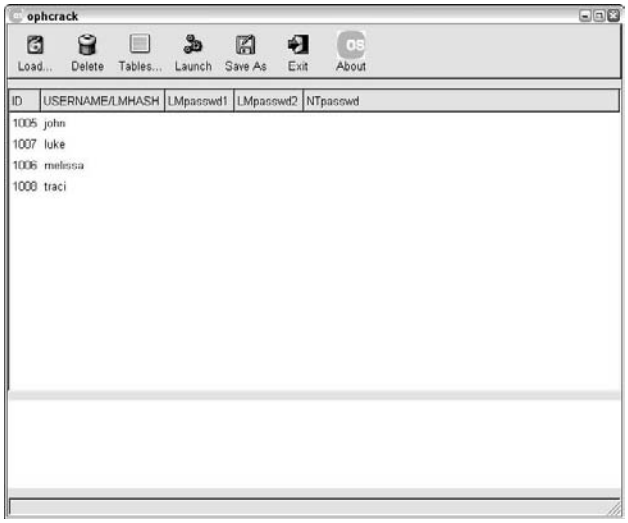


Figure 7-5: Usernames extracted via ophcrack.

5. **Click the Launch icon to begin the rainbow crack process.**

   The process can take a little while depending on your computer's speed. Three of the long, random passwords I created for my test accounts were cracked in just a couple of minutes, as shown in Figure 7-6. The only reason the fourth wasn't cracked is because it had an exclamation point on the end and I was using ophcrack's smaller "10k" alphanumeric character set that doesn't test for extended characters. ophcrack has other options that will test for extended characters, so no worries for more "creative" passwords.

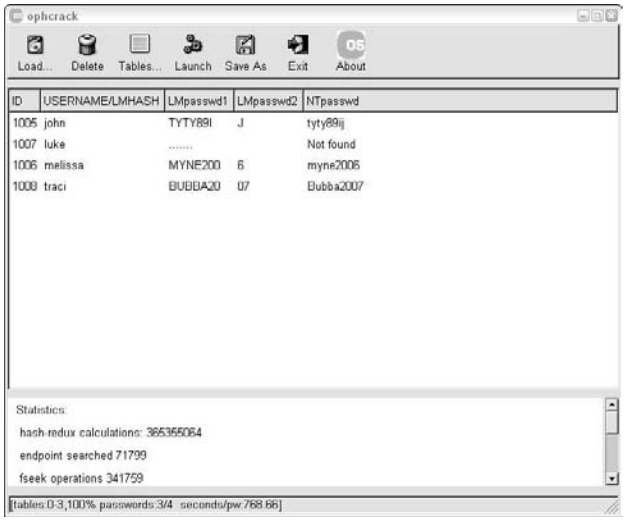


Figure 7-6: Cracked hashes using ophcrack.

There's also a bootable Linux-based version of ophcrack (available at `http://ophcrack.sourceforge.net`) that allows you to boot a system and start cracking passwords without having to log in or install any software.

I *highly* recommend you use the ophcrack LiveCD on a sample laptop computer or two to demonstrate just how simple it is to recover passwords, and subsequently, sensitive information from laptops that do not have have encrypted hard drives.

Before submitting password hashes to a third party, make sure doing so will not violate any internal policies, business contracts, or nondisclosure agreements or get you into hot water. Also, remember that submitting password hashes to a third party creates an accountability issue because three or more parties technically have access to the passwords.

Checking for null/blank passwords in NetWare

By using the chknull program, you can test for NetWare users that have empty passwords, passwords that match their usernames, and passwords that match a specific password that you supply on the command line. Figure 7-7 shows the output of a chknull session against a NetWare server without being logged in: Four users have blank passwords, three users have the password "123," and one user's password is the same as her username (`avadminuser`).

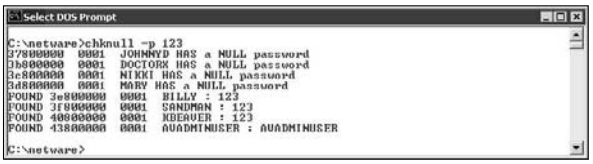


Figure 7-7: NetWare password weaknesses found with chknull.

Password-protected files

Do you wonder how vulnerable password-protected word-processing, spreadsheet, and zip files are when users send them into the wild blue yonder? Wonder no more. Some great utilities can show how easily passwords are cracked.

Cracking files

Most password-protected files can be cracked in seconds or minutes. You can demonstrate this "wow factor" security vulnerability to users and management. Here's a hypothetical real-world scenario:

1. Your CFO wants to send some confidential financial information in an Excel spreadsheet to a company board member.
2. She protects the spreadsheet by assigning it a password during the file-save process in Excel.
3. For good measure, she uses WinZip to compress the file, and adds another password to make it *really* secure.
4. The CFO sends the spreadsheet as an e-mail attachment, assuming that the e-mail will reach its destination.

   The financial advisor's network has content filtering, which monitors incoming e-mails for keywords and file attachments. Unfortunately, the financial advisory firm's network administrator is looking in the content-filtering system to see what's coming in.
5. This rogue network administrator finds the e-mail with the confidential attachment, saves the attachment, and realizes that it's password protected.
6. The network administrator remembers a great password-cracking tool available from Elcomsoft called Advanced Archive Password Recovery (`www.elcomsoft.com/archpr.html`) that can help him out so he proceeds to use it to crack the password.

Cracking password-protected files is as simple as that! Now all that the rogue network administrator must do is forward the confidential spreadsheet to his buddies or to the company's competitors.

If you carefully select the right options in Advanced Archive Password Recovery, you can drastically shorten your testing time. For example, if you know that a password is not over five characters long or is lowercase letters only, you can cut the cracking time in half.

I recommend performing these file password-cracking tests on files that you capture with a content filtering or network analysis tool. This is a good way to determine whether your users are adhering to policy and using adequate passwords to protect sensitive information they're sending.

Countermeasures

The best defense against weak file password protection is to require your users to use a stronger form of file protection, such as PGP, or the AES encryption that's built in to WinZip, when necessary. Ideally, you don't want to rely on users to make decisions about what they should use to secure sensitive information, but it's better than nothing. Stress that a file encryption mechanism, such as a password-protected zip file, is secure only if users keep their passwords confidential and never transmit or store them in unsecure cleartext (such as in a separate e-mail).

If you're concerned about unsecure transmissions through e-mail, consider using a content filtering or data leak prevention system to block all outbound e-mail attachments that aren't protected on your e-mail server.

Other ways to crack passwords

Over the years, I've found other ways to crack (or capture) passwords technically and through social engineering.

Keystroke logging

One of the best techniques for capturing passwords is remote *keystroke logging* — the use of software or hardware to record keystrokes as they're typed into the computer.

Be careful with keystroke logging. Even with good intentions, monitoring employees raises various legal issues if it's not done correctly. Discuss with your legal counsel what you'll be doing, ask for their guidance, and get approval from upper management.

Logging tools

With keystroke-logging tools, you can assess the log files of your application to see what passwords people are using:

- Keystroke-logging applications can be installed on the monitored computer. I recommend that you check out eBlaster and Spector Pro by SpectorSoft (`www.spectorsoft.com`). Another popular tool is Invisible KeyLogger Stealth, available at `www.amecisco.com/iks.htm`. Dozens of other such tools are available on the Internet.
- Hardware-based tools, such as KeyGhost (`www.keyghost.com`), fit between the keyboard and the computer or replace the keyboard altogether.

A keystroke-logging tool installed on a shared computer can capture the passwords of every user who logs in.

Countermeasures

The best defense against the installation of keystroke-logging software on your systems is to use a spyware-detection program or other antivirus product. As for physical keyloggers, you'll need to visually inspect each system.

The potential for hackers to install keystroke-logging software is another reason to ensure that your users aren't downloading and installing random shareware or opening attachments in unsolicited e-mails. Consider locking down your desktops by setting the appropriate user rights through local or group security policy in Windows. Alternatively, you could use a commercial lockdown program, such as Fortres 101 (`www.fortresgrand.com`) for Windows or Deep Freeze (`www.faronics.com/html/deepfreeze.asp`) for Windows, Linux, and Mac OS X.

Weak password storage

Many legacy and standalone applications, such as e-mail, dial-up network connections, and accounting software, store passwords locally, making them vulnerable to password hacking. By performing a basic text search, I've found passwords stored in cleartext on the local hard drives of machines. You can automate the process even further by using a program called Identity Finder Pro (`www.identityfinder.com/pro`). I cover these file and related storage vulnerabilities in Chapter 15.

Searching

You can try using your favorite text-searching utility — such as the Windows search function, `findstr`, or `grep` — to search for *password* or *passwd* on your computer's drives. You might be shocked to find what's on your systems. Some programs even write passwords to disk or leave them stored in memory.

Weak password storage is a hacker's dream. Head it off if you can.

Countermeasures

The only reliable way to eliminate weak password storage is to use only applications that store passwords securely. This might not be practical, but it's your only guarantee that your passwords are secure. Another option is to instruct users not to store their passwords when prompted.

Before upgrading applications, contact your software vendor to see how they manage passwords, or search for a third-party solution.

Network analyzer

A network analyzer sniffs the packets traversing the network. This is what the bad guys do if they can gain control of a computer, tap into your wireless network, or gain physical network access to set up their network analyzer. If they gain physical access, they can look for a network jack on the wall and plug right in!

Testing

Figure 7-8 shows how crystal-clear passwords can be through the eyes of a network analyzer. This figure shows how Cain & Abel (`www.oxid.it/cain.html`) can glean thousands of passwords going across the network in a matter of a couple of hours. As you can see in the left pane, these clear-text password vulnerabilities can apply to FTP, Web, telnet, and more. (The actual usernames and passwords are blurred out to protect them.)

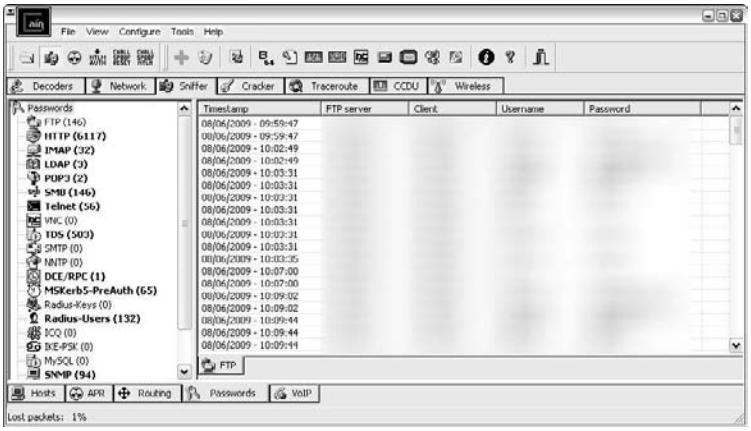


Figure 7-8: Using Cain & Abel to capture passwords going across the network.

If traffic is not tunneled through a VPN, SSH, SSL, or some other form of encrypted link, it's vulnerable to attack.

Cain & Abel is a password-cracking tool that also has network analysis capabilities. You can also use a regular network analyzer, such as the commercial products OmniPeek (`www.wildpackets.com/products/distributed_network_analysis/omnipeek_network_analyzer`) and CommView (`www.tamos.com/products/commview`) as well as the free open source program, Wireshark (`www.wireshark.org`). With a network analyzer, you can search for password traffic in various ways. For example, to capture POP3 password traffic, you can set up a filter and a trigger to search for the PASS command. When the network analyzer sees the PASS command in the packet, it captures that specific data.

Network analyzers require you to capture data on a hub segment of your network or via a monitor/mirror/span port on a switch. Otherwise, you can't see anyone else's data traversing the network — just yours. Check your switch's user guide for whether it has a monitor or mirror port and instructions on how to configure it. You can connect your network analyzer to a hub on the public side of your firewall. You'll capture only those packets that are entering or leaving your network — not internal traffic. I cover this type of network infrastructure hacking in detail in Chapter 8.

Countermeasures

Here are some good defenses against network analyzer attacks:

- **Use switches on your network, not hubs.** If you must use hubs on network segments, a program like sniffdet (`http://sniffdet.sourceforge.net`) for UNIX-based systems and PromiscDetect (`http://ntsecurity.nu/toolbox/promiscdetect`) for Windows can detect network cards in *promiscuous mode* (accepting all packets, whether destined for the local machine or not). A network card in promiscuous mode signifies a network analyzer is running on the network.
- **Make sure that unsupervised areas, such as an unoccupied lobby or training room, don't have live network connections.**
- **Don't let anyone without a business need gain physical access to your switches or the network connection on the public side of your firewall.** With physical access, a hacker can connect to a switch monitor port or tap into the unswitched network segment outside the firewall and capture packets.

Switches do not provide complete security because they are vulnerable to ARP poisoning attacks, which I cover in Chapter 8.

Weak BIOS passwords

Most computer BIOS (basic input/output system) settings allow power-on passwords and/or setup passwords to protect the computer's hardware settings that are stored in the CMOS chip. Here are some ways around these passwords:

- You can usually reset these passwords either by unplugging the CMOS battery or by changing a jumper on the motherboard.
- Password-cracking utilities for BIOS passwords are available on the Internet and from computer manufacturers.
- If gaining access to the hard drive is your ultimate goal, you can simply remove the hard drive from the computer and install it in another one and you're good to go. This is a great way to prove that BIOS/power-on passwords are *not* an effective countermeasure for lost or stolen laptops.

For a good list of default system passwords for various vendor equipment, check `www.cirt.net/passwords`.

Countermeasures

There are tons of variables for hacking and hacking countermeasures depending on your hardware setup. If you plan to hack your own BIOS passwords, check for information in your user manual or refer to the BIOS password hacking guide I wrote at `http://tinyurl.com/fwom6`. If protecting the information on your hard drives is your ultimate goal then whole disk (PGP) or whole volume (Windows BitLocker) encryption is the best way to go.

Weak passwords in limbo

Bad guys often exploit user accounts that have just been created or reset by a network administrator or help desk. New accounts might need to be created for new employees or even for your own ethical hacking purposes. Accounts might need to be reset if users forget their passwords or if the accounts have been locked out because of failed attempts.

Weaknesses

Here are some reasons why user accounts can be vulnerable:

- When user accounts are reset, they often are assigned an easily cracked password (such as the user's name or the word *password*). The time between resetting the user account and changing the password is a prime opportunity for a break-in.
- Many systems have either default accounts or unused accounts with weak passwords or no passwords at all. These are prime targets.

Countermeasures

The best defenses against attacks on passwords in limbo are solid help desk policies and procedures that prevent weak passwords from being available at *any* given time during the new account generation and password reset processes. Perhaps the best ways to overcome this vulnerability are as follows:

- Require users to be on the phone with the help desk, or have a help desk member perform the reset at the user's desk.
- Require that the user immediately log in and change the password.
- If you need the ultimate in security, implement stronger authentication methods, such as challenge/response questions, smart cards, or digital certificates.
- Automate password reset functionality on your network so users can manage most of their password problems without help from others.

Password-reset programs

Network administrators occasionally use programs that reset the administrator password, which can be used against a network.

Tools

My favorite tool for this task is Elcomsoft System Recovery (`www.elcomsoft.com/esr.html`). You simply burn this tool to a CD and use it to boot the system you want to recover the password from, as shown in Figure 7-9.

You can also use another proven tool for Windows called NTAccess (`www.mirider.com/ntaccess.html`). This program isn't pretty or fancy, but it does the job. As with ophcrack, these tools provide an excellent way to demonstrate that you need to encrypt your laptop hard drives.

If you want to perform similar checks on a UNIX or Linux-based laptop, you should be able to boot from a Knoppix (`www.knoppix.net`) or similar "live" Linux distribution and edit the local passwd file (`/etc/shadow`) to reset or change it. Remove the encrypted code between the first and second colons for the "root" (or whatever user) entry or copy the password from the entry of another user and paste it into that area.

Countermeasures

The best safeguard against a hacker using a password reset program against your Windows systems is to encrypt your hard drives by using Windows BitLocker in Windows Vista and Windows 7 or PGP Whole Disk Encryption (`www.pgp.com/products/wholediskencryption`). For Linux you can use TrueCrypt (`www.truecrypt.org`). You also need to ensure that people can't gain unauthorized physical access to your computers. When a hacker has physical access and your drives are not encrypted, all bets are off.

Figure 7-9: Elcomsoft System Recovery CD for resetting Windows passwords.

General Password-Cracking Countermeasures

A password for one system usually equals passwords for many other systems because many people use the same (or at least similar) passwords on every system they use. For this reason, you might want to consider instructing users to create different passwords for different systems, especially on the systems that protect information that's more sensitive. The only downside to this is that users have to keep multiple passwords and, therefore, might be tempted to write them down, which can negate any benefits.

Strong passwords are important, but balance security and convenience:

- You can't expect users to memorize passwords that are insanely complex and must be changed every few weeks.
- You can't afford weak passwords or no passwords at all, so come up with a strong password policy and accompanying standard — preferably one that requires long and strong passphrases (combinations of words that are easily remembered yet next to impossible to crack) that have to be changed only once or twice a year.

Storing passwords

If you have to choose between weak passwords that your users can memorize and strong passwords that your users must write down, I recommend having readers write down passwords and store the information securely. Train users to store their written passwords in a secure place — not on keyboards or in easily cracked password-protected computer files (such as spreadsheets). Users should store a written password in either of these locations:

- A locked file cabinet or office safe
- An encrypted file or database, using such tools as
  - PGP (`www.pgpi.org` offers the free, open-source version, and `www.pgp.com` offers the commercial version)
  - Password Safe, an open-source software originally developed by Counterpane (`http://passwordsafe.sourceforge.net`)

No passwords on sticky notes! People joke about it, but it happens a lot and it's not good for business!

Policy considerations

As an ethical hacker, you should show users the importance of securing their passwords. Here are some tips on how to do that:

- **Demonstrate how to create secure passwords.** Refer to them as *pass-phrases* because people tend to take *passwords* literally and use only words, which can be less secure.
- **Show what can happen when weak passwords are used or passwords are shared.**
- **Diligently build user awareness of social engineering attacks.**

Enforce (or at least encourage the use of) a strong password-creation policy that includes the following criteria:

- **Use upper- and lowercase letters, special characters, and numbers.** Never use only numbers. These passwords can be cracked quickly.
- **Misspell words or create acronyms from a quote or a sentence.** For example, *ASCII* is an acronym for *American Standard Code for Information Interchange* that can also be used as part of a password.
- **Use punctuation characters to separate words or acronyms.**

- **Change passwords every 6 to 12 months or immediately if they're suspected of being compromised.** Anything more frequent introduces an inconvenience that only serves to create more vulnerabilities.
- **Use different passwords for each system.** This is especially important for network infrastructure hosts, such as servers, firewalls, and routers. It's okay to use similar passwords — just make them slightly different for each type of system, such as *SummerInTheSouth_WinXP* for Windows systems and *SummerInTheSouth_Lin* for Linux systems.
- **Use variable-length passwords.** This can throw off attackers because they won't know the required minimum or maximum length of passwords and must try all password length combinations.
- **Don't use common slang words or words that are in a dictionary.**
- **Don't rely completely on similar-looking characters, such as *3* instead of *E, 5* instead of *S,* or *!* instead of *1*.** Password cracking programs can check for this.
- **Don't reuse the same password within at least four to five password changes.**
- **Use password-protected screen savers.** Unlocked screens are a great way for systems to be compromised even if their hard drives are encrypted.
- **Don't share passwords.** To each his or her own!
- **Avoid storing user passwords in an unsecured central location, such as an unprotected spreadsheet on a hard drive.** This is an invitation for disaster. Use PGP, Password Safe, or a similar program to store user passwords.

Other considerations

Here are some other password-hacking countermeasures that I recommend:

- **Enable security auditing to help monitor and track password attacks.**
- **Test your applications to make sure they aren't storing passwords indefinitely in memory or writing them to disk.** A good tool for this is WinHex (`www.winhex.com/winhex/index-m.html`). I've used this tool to search a computer's memory for *password, pass=, login,* and so on and have come up with some passwords that the developers thought were cleared from memory.

  Some password-cracking Trojan-horse applications are transmitted through worms or simple e-mail attachments. Such malware can be lethal to your password-protection mechanisms if they're installed on

your systems. The best defense is malware protection software, such as antivirus protection (from a vendor like Webroot or McAfee), spyware protection (such as Spybot), or malicious-code behavioral protection (such as Finjan's offerings).

- **Keep your systems patched.** Passwords are reset or compromised during buffer overflows or other denial of service (DoS) conditions.
- **Know your user IDs.** If an account has never been used, delete or disable the account until it's needed. You can determine unused accounts by manual inspection or by using DumpSec (`www.systemtools.com/somarsoft/?somarsoft.com`), a tool that can enumerate the Windows operating system and gather user IDs and other information.

As the security administrator in your organization, you can enable *account lockout* to prevent password-cracking attempts. Account lockout is the ability to lock user accounts for a certain time after a certain number of failed login attempts has occurred. Most operating systems (and some applications) have this capability. Don't set it too low (fewer than five failed logins), and don't set it too high to give a malicious user a greater chance of breaking in. Somewhere between 5 and 50 might work for you. I usually recommend a setting of around 10 or 15. Consider the following when configuring account lockout on your systems:

- To use account lockout to prevent any possibilities of a user DoS condition, require two different passwords, and don't set a lockout time for the first one if that feature is available in your operating system.
- If you permit autoreset of the account after a certain period — often referred to as *intruder lockout* — don't set a short time period. Thirty minutes often works well.

A failed login counter can increase password security and minimize the overall effects of account lockout if the account experiences an automated attack. A login counter can force a password change after a number of failed attempts. If the number of failed login attempts is high and occurred over a short period, the account has likely experienced an automated password attack.

Other password-protection countermeasures include

- **Stronger authentication methods,** such as challenge/response, smart cards, tokens, biometrics, or digital certificates.
- **Automated password reset.** This functionality lets users manage most of their password problems without getting others involved. Otherwise, this support issue becomes expensive, especially for larger organizations.
- **Password protect the system BIOS.** This is especially important on servers and laptops that are susceptible to physical security threats and vulnerabilities.

Securing Operating Systems

You can implement various operating system security measures to ensure that passwords are protected.

Regularly perform these low-tech and high-tech password-cracking tests to make sure that your systems are as secure as possible — perhaps as part of a monthly, quarterly, or biannual audit.

Windows

The following countermeasures can help prevent password hacks on Windows systems:

- Some Windows passwords can be gleaned by simply reading the cleartext or crackable ciphertext from the Windows Registry. Secure your registries by doing the following:
  - Allow only administrator access.
  - Harden the operating system by using well-known hardening best practices, such as those from SANS (`www.sans.org`), NIST (`http://csrc.nist.gov`), and the Center for Internet Security Benchmarks/Scoring Tools (`www.cisecurity.org`), and the ones outlined in *Network Security For Dummies,* by Chey Cobb.
- Use SYSKEY for enhanced Windows password protection.
  - By default, Windows 2000 and newer systems encrypts the SAM database that stores hashes of the Windows account passwords. Encryption is not the default on older Windows NT systems.
  - You can use the SYSKEY utility to encrypt the database for Windows NT machines and to move the database encryption key from Windows 2000 and later machines.

  Don't rely on only the SYSKEY utility. Many tools can crack SYSKEY encryption.
- Keep all SAM database backup copies secure.
- Disable the storage of LM hashes in Windows for passwords that are shorter than 15 characters.

  For example, in Windows 2000 SP2 and later, you can create and set the NoLMHash registry key to a value of 1 under `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa`.

- Use `passfilt.dll` or local or group security policies to help eliminate weak passwords on Windows systems before they're created.
- Disable null sessions in your Windows version:
  - In Windows XP and later versions, enable the Do Not Allow Anonymous Enumeration of SAM Accounts and Shares option in the local security policy.
  - In Windows 2000, enable the No Access without Explicit Anonymous Permissions option in the local security policy.
  - In Windows NT, enable the following Registry key:

```
HKLM/System/CurrentControlSet/Control/LSA/
RestrictAnonymous=1
```

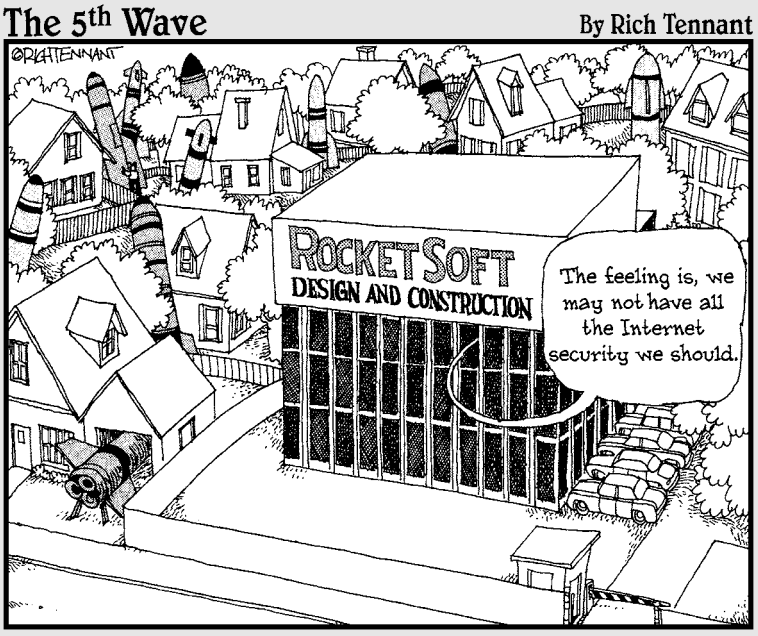
Linux and UNIX

The following countermeasures can help prevent password cracks on Linux and UNIX systems:

- Ensure your system is using shadowed MD5 passwords.
- Help prevent the creation of weak passwords. You can use either the built-in operating-system password filtering (such as cracklib in Linux) or a password-auditing program (such as npasswd or passwd+).
- Check your `/etc/passwd` file for duplicate root UID entries. Hackers can exploit such entries as root backdoors.

Part III

Hacking the Network



In this part . . .

Now that you're off and running with your ethical hacking tests, it's time to take things to a new level. The tests in the previous part — at least the social engineering and physical security tests — start at a high level and are not that technical. Times, they are a-changin'! You now need to look at network security. This is where things start getting more involved.

This part starts out by looking at the network from the inside and the outside for everything from perimeter security holes to network device exploits to DoS vulnerabilities and more. This part then looks at how to assess the security of wireless LANs that introduce some serious security vulnerabilities into networks these days.

Chapter 8

Network Infrastructure

In This Chapter

- Selecting tools
- Scanning network hosts
- Assessing security with a network analyzer
- Preventing denial-of-service and infrastructure vulnerabilities

To have secure operating systems and applications, you need to have a secure network. Devices such as routers, firewalls, and even generic hosts (including servers and workstations) must be assessed as part of the ethical hacking process.

There are thousands of possible network vulnerabilities, equally as many tools, and even more testing techniques. You probably don't have the time or resources available to test your network infrastructure systems for *all* possible vulnerabilities, using every tool and method imaginable. Instead, you need to focus on tests that will produce a good overall assessment of your network — and the tests I describe in this chapter produce exactly that.

You can eliminate many well-known, network-related vulnerabilities by simply patching your network hosts with the latest vendor software and firmware updates. Because most network infrastructure hosts are not publicly accessible, odds are that your network hosts *will not* be attacked from the outside and even when they are, the results are not likely to be detrimental. You can eliminate many other vulnerabilities by following some solid security practices on your network, as described in this chapter and in *Network Security Bible,* Second Edition by Eric Cole. The tests, tools, and techniques outlined in this chapter offer the most bang for your ethical hacking buck.

The better you understand network protocols, the easier network vulnerability testing is because network protocols are the foundation for most information security concepts. If you're a little fuzzy on how networks work, I highly encourage you to read *TCP/IP For Dummies,* 6th Edition, by Candace Leiden and Marshall Wilensky and the Request for Comments (RFCs) list on the Official Internet Protocol Standards page, `www.rfc-editor.org/rfcxx00.html`.

A case study in hacking network infrastructures with Laura Chappell

Laura Chappell — one of the world's foremost authorities on network protocols and analysis — shared with me an interesting experience she had when assessing a customer's network.

The Situation

A customer called Ms. Chappell with a routine "the network is slow" problem. Upon Ms. Chappell arrival onsite, the customer mentioned sporadic outages and poor performance when connecting to the Internet. First, Ms. Chappell examined individual flows between various clients and servers. Localized communications appeared normal, but any communication that flowed through the firewall to the Internet or other branch offices was severely delayed. Ms. Chappell sniffed the traffic going through the firewall to see whether she could isolate the cause of the delay.

The Outcome

A quick review of the traffic crossing the firewall indicated that the outside links were saturated, so Ms. Chappell needed to review and classify the traffic. Using the network analyzer, Ms. Chappell plugged in to examine the protocol distribution. She saw that almost 45 percent of the traffic was listed as "others" and was unrecognizable. Laura captured some data and found several references to pornographic images. Further examination of the packets led her to two specific port numbers that appeared consistently in the trace files — ports 1214 (Kazaa) and 6346 (Gnutella), two peer-to-peer (P2P) file-sharing applications. Ms. Chappell did a complete port scan of the network to see what was running and found more than 30 systems running either Kazaa or Gnutella. Their file transfer processes were eating up the bandwidth and dragging down all communications. Shutting down these systems and removing the applications would have been simple, but Laura wanted to investigate them further without the users' knowledge.

Ms. Chappell decided to use her own Kazaa and Gnutella clients to look through the shared folders of the systems. By becoming a peer member with the other hosts on the network, Ms. Chappell could perform searches through other shared folders, which indicated some of the users had shared their network directories. Through these shared folders, Ms. Chappell obtained the corporate personnel roster, including home phone numbers and addresses, accounting records, and several confidential memos that provided timelines for projects at the company.

Many users said they shared these folders to regain access to the P2P network because they had been labeled *freeloaders* — their shares contained only a few files. They were under the delusion that because no one outside the company knew the filenames contained in the network directories, a search wouldn't come up with matching values, and no one would download those files. Although this onsite visit started with a standard performance and communication review, it ended with the detection of some huge security breaches in the company. Anyone could have used these P2P tools to get onto the network and grab the files in the shared folders — with no authorization or authentication required.

Laura Chappell is Senior Protocol Analyst at the Protocol Analysis Institute, LLC (`www.packet-level.com`). A best-selling author and lecturer, Ms. Chappell has trained thousands of network administrators, security technicians, and law enforcement personnel on packet-level security, troubleshooting, and optimization techniques. I *highly* recommend that you check out her Web site for some excellent technical content that can help you become a better ethical hacker.

Network Infrastructure Vulnerabilities

Network infrastructure vulnerabilities are the foundation for all technical security issues in your information systems. These lower-level vulnerabilities affect everything running on your network. That's why you need to test for them and eliminate them whenever possible.

Your focus for ethical hacking tests on your network infrastructure should be to find weaknesses that others can see in your network so you can quantify your network's level of exposure.

Many issues are related to the security of your network infrastructure. Some issues are more technical and require you to use various tools to assess them properly. You can assess others with a good pair of eyes and some logical thinking. Some issues are easy to see from outside the network, and others are easier to detect from inside your network.

When you assess your company's network infrastructure security, you need to look at as the following:

- Where devices, such as a firewall or IPS, are placed on the network and how they're configured
- What external attackers see when they perform port scans, and how they can exploit vulnerabilities in your network hosts
- Network design, such as Internet connections, remote access capabilities, layered defenses, and placement of hosts on the network
- Interaction of installed security devices, such as firewalls, IPSes, antivirus, and so on
- What protocols are in use
- Commonly attacked ports that are unprotected
- Network host configurations
- Network monitoring and maintenance

If someone exploits a vulnerability in one of the items in the preceding list or anywhere in your network's security, bad things can happen:

- A hacker can use a denial of service (DoS) attack, which can take down your Internet connection — or even your entire network.
- A malicious employee using a network analyzer can steal confidential information in e-mails and files sent on the network.
- A hacker can set up backdoors into your network.
- A hacker can attack specific hosts by exploiting local vulnerabilities across the network.

Before assessing your network infrastructure security, remember to do the following:

- Test your systems from the outside in, the inside out, and the inside in (that is, between internal network segments and demilitarized zones [DMZs]).
- Obtain permission from partner networks to check for vulnerabilities on their ends that can affect *your* network's security, such as open ports, lack of a firewall, or a misconfigured router.

Choosing Tools

Your tests require the right tools — you need scanners, analyzers, and vulnerability assessment tools. Great commercial, shareware, and freeware tools are available. I describe a few of my favorite tools in the following sections. Just keep in mind that you need more than one tool, and that no tool does everything you need.

If you're looking for easy-to-use security tools with all-in-one packaging, you get what you pay for most of the time — especially for the Windows platform. Tons of security professionals swear by many free security tools, especially those that run on Linux and other UNIX-based operating systems. Many of these tools offer a lot of value — if you have the time, patience, and willingness to learn their ins and outs.

Scanners and analyzers

These scanners provide practically all the port scanning and network testing you need:

- **SuperScan** (`www.foundstone.com/us/resources/proddesc/superscan.htm`) for ping sweeps and port scanning
- **Essential NetTools** (`www.tamos.com/products/nettools`) for a wide variety of network scanning functionality
- **NetScanTools Pro** (`www.netscantools.com`) for dozens of network security assessment functions, including ping sweeps, port scanning, and SMTP relay testing
- **Getif** (`www.wtcs.org/snmp4tpc/getif.htm`) for SNMP enumeration
- **Nmap** (`www.insecure.org/nmap`) — or **NMapWin** (`http://sourceforge.net/projects/nmapwin`), the happy-clicky-GUI front end to Nmap — for host-port probing and operating system fingerprinting

- **Cain & Abel** (`www.oxid.it/cain.html`) for network analysis and ARP poisoning
- **WildPackets' OmniPeek** (`www.wildpackets.com/products/distributed_network_analysis/omnipeek_network_analyzer`) for network analysis

Vulnerability assessment

These vulnerability assessment tools allow you to test your network hosts for various known vulnerabilities as well as potential configuration issues that could lead to security exploits:

- **GFI LANguard** (`www.gfi.com/lannetscan`) for port scanning and vulnerability testing
- **Nessus** (`www.nessus.org`), a free all-in-one tool for ping sweeps, port scanning, and vulnerability testing
- **QualysGuard** (`www.qualys.com`), a great all-in-one tool for in-depth vulnerability testing

Scanning, Poking, and Prodding

Performing the ethical hacks described in the following sections on your network infrastructure involves following basic hacking steps:

1. Gather information and map your network.
2. Scan your systems to see which ones are available.
3. Determine what's running on the systems discovered.
4. Attempt to penetrate the systems discovered, if you choose to.

Every network card driver and implementation of TCP/IP in most operating systems, including Windows and Linux, and even in your firewalls and routers, has quirks that result in different behaviors when scanning, poking, and prodding your systems. This can result in different responses from your various systems, including everything from false-positive findings to denial of service (DoS) conditions. Refer to your administrator guides or vendor Web sites for details on any known issues and possible patches that are available to fix them. If you have all your systems patched, this shouldn't be an issue.

Port scanners

A port scanner shows you what's what on your network. A software tool that scans the network to see what's alive and working, port scanners provide basic views of how the network is laid out. They can help identify unauthorized hosts or applications and network host configuration errors that can cause serious security vulnerabilities.

The big-picture view from port scanners often uncovers security issues that might otherwise go unnoticed. Port scanners are easy to use and can test systems regardless of what operating systems and applications they're running. The tests are usually performed relatively quickly without having to touch individual network hosts, which would be a real pain otherwise.

The trick to assessing your overall network security is interpreting the results you get from a port scan. You can get false positives on open ports, and you might have to dig deeper. For example, UDP scans — like the protocol itself — are less reliable than TCP scans and often produce false positives because many applications don't know how to respond to random incoming UDP requests.

A feature-rich scanner often can identify ports and see what's running in one step.

Port scans can take a good bit of time. The length of time depends on the number of hosts you have, the number of ports you scan, the tools you use, the processing power of your test system, and the speed of your network links.

An important tenet to remember is that you need to scan more than just the important hosts. Leave no stone unturned. These other systems often bite you if you ignore them. Also, perform the same tests with different utilities to see whether you get different results. Not all tools find the same open ports and vulnerabilities. This is unfortunate, but it's a reality of ethical hacking tests.

If your results don't match after you run the tests using different tools, you might want to explore the issue further. If something doesn't look right — such as a strange set of open ports — it probably isn't. Test again; if you're in doubt, use another tool for a different perspective.

As an ethical hacker, you should scan all 65,535 TCP ports on each network host that your scanner finds. If you find questionable ports, look for documentation that the application is known and authorized. It's not a bad idea to scan all 65,535 UDP ports as well.

For speed and simplicity, you can scan the commonly hacked ports, listed in Table 8-1.

Table 8-1 Commonly Hacked Ports

Port Number	*Service*	*Protocol(s)*
7	Echo	TCP, UDP
19	Chargen	TCP, UDP
20	FTP data (File Transfer Protocol)	TCP
21	FTP control	TCP
22	SSH	TCP
23	Telnet	TCP
25	SMTP (Simple Mail Transfer Protocol)	TCP
37	Daytime	TCP, UDP
53	DNS (Domain Name System)	UDP
69	TFTP (Trivial File Transfer Protocol)	UDP
79	Finger	TCP, UDP
80	HTTP (Hypertext Transfer Protocol)	TCP
110	POP3 (Post Office Protocol version 3)	TCP
111	SUN RPC (remote procedure calls)	TCP, UDP
135	RPC/DCE (end point mapper) for Microsoft networks	TCP, UDP
137, 138, 139, 445	NetBIOS over TCP/IP	TCP, UDP
161	SNMP (Simple Network Management Protocol)	TCP, UDP
443	HTTPS (HTTP over SSL)	TCP
512, 513, 514	Berkeley r-services and r-commands (such as `rsh`, `rexec`, and `rlogin`)	TCP
1433	Microsoft SQL Server (ms-sql-s)	TCP, UDP
1434	Microsoft SQL Monitor (ms-sql-m)	TCP, UDP
1723	Microsoft PPTP VPN	TCP
3389	Windows Terminal Server	TCP
5631, 5632	pcAnywhere	TCP
8080	HTTP proxy	TCP

Ping sweeping

A ping sweep of all your network subnets and hosts is a good way to find out which hosts are alive and kicking on the network. A *ping sweep* is when you ping a range of addresses using Internet Control Message Protocol (ICMP) packets. Figure 8-1 shows the command and the results of using Nmap to perform a ping sweep of a class C subnet range.

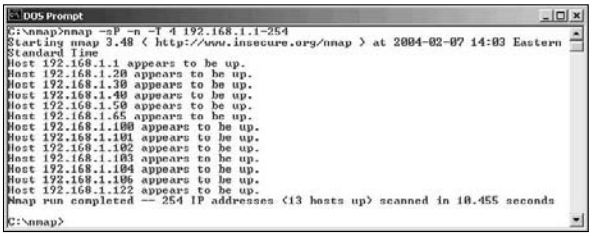


Figure 8-1: Performing a ping sweep of an entire class C network with Nmap.

Dozens of Nmap command-line options exist, which can be overwhelming when you want only a basic scan. Nonetheless, you can enter `nmap` on the command line to see all the options available.

The following command-line options can be used for an Nmap ping sweep:

- `-sP` tells Nmap to perform a ping scan.
- `-n` tells Nmap not to perform name resolution.

  You can omit the `-n` option if you want to resolve hostnames to see which systems are responding. Name resolution might take slightly longer, though.
- `-T 4` tells Nmap to perform an aggressive (faster) scan.
- `192.168.1.1-254` tells Nmap to scan the entire 192.168.1.x subnet.

Using port scanning tools

Most port scanners operate in three steps:

1. The port scanner sends TCP SYN requests to the host or range of hosts you set it to scan.

   Some port scanners, such as SuperScan, perform ping sweeps to determine which hosts are available before starting the TCP port scans.

   Most port scanners by default scan only TCP ports. Don't forget about UDP ports. You can scan UDP ports with a UDP port scanner, such as Nmap.
2. The port scanner waits for replies from the available hosts.
3. The port scanner probes these available hosts for up to 65,535 possible TCP and UDP ports — based on which ports you tell it to scan — to see which ones have available services on them.

The port scans provide the following information about the live hosts on your network:

- Hosts that are active and reachable through the network
- Network addresses of the hosts found
- Services or applications that the hosts *may be* running

After performing a generic sweep of the network, you can dig deeper into specific hosts you find.

SuperScan

My favorite tool for performing generic TCP port scans is SuperScan version 3.0. Don't laugh because it's so old! It's reliable, which goes a long way in my book. Figure 8-2 shows the results of my scan and a few interesting ports open on several hosts, including Windows Terminal Server and SSH.

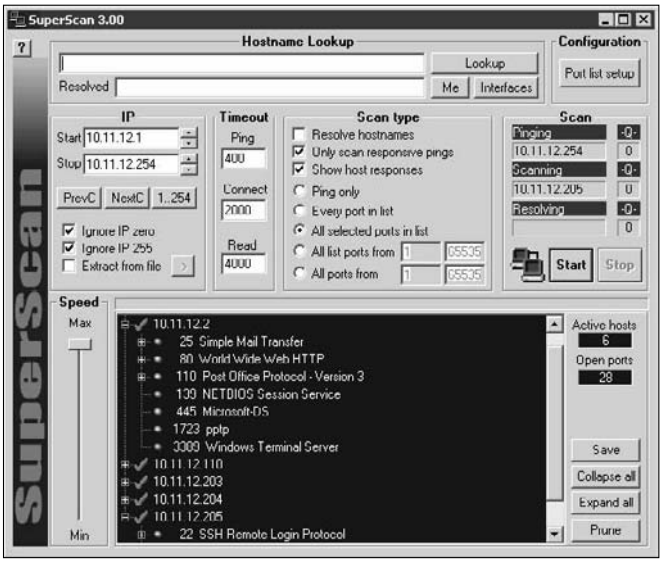


Figure 8-2: A TCP port scan using SuperScan version 3.0.

In Figure 8-2, I selected the Only Scan Responsive Pings and All Selected Ports in List options. However, you might want to select some other options:

- If you don't want to ping each host first, deselect the Only Scan Responsive Pings option. ICMP can be blocked, which can cause the scanner not to find certain hosts, so this option can make the test run more efficiently.
- If you want to scan a certain range of well-known ports or ports specific to your systems, you can configure SuperScan to do so. I recommend these settings:
  - If you want to perform a scan on well-known ports, at least select the All Selected Ports in List option.
  - If this is your initial scan, scan all ports from 1 to 65,535.

Nmap

After you have a general idea of what hosts are available and what ports are open, you can perform fancier scans to verify that the ports are actually open and not returning a false positive. If you wish to do this, Nmap is the perfect tool to use. Nmap allows you to run the following additional scans:

- **Connect:** This basic TCP scan looks for any open TCP ports on the host. You can use this scan to see what's running and determine whether intrusion detection systems (IDSes), intrusion prevention systems (IPSes), firewalls, or other logging devices log the connections.
- **UDP scan:** This basic UDP scan looks for any open UDP ports on the host. You can use this scan to see what's running and determine whether IPSes, firewalls, or other logging devices log the connections.
- **SYN Stealth:** This scan creates a half-open TCP connection with the host possibly evading IDS systems and logging. This is a good scan for testing IDSes, firewalls, and other logging devices.
- **FIN Stealth, Xmas Tree, and Null:** These scans let you mix things up a bit by sending strangely formed packets to your network hosts so you can see how they respond. These scans change around the flags in the TCP headers of each packet, which allows you to test how each host handles them to point out weak TCP/IP implementations and patches that might need to be applied.

Be careful when performing these scans. You can create your own DoS attack and potentially crash applications or entire systems. Unfortunately, if you have a host with a weak TCP/IP stack (the software that controls TCP/IP communications on your hosts), there's no good way to prevent your scan from creating a DoS attack. A good way to help reduce the chance of this occurring is to use the slow Nmap timing options — Paranoid, Sneaky, or Polite — when running your scans.

Figure 8-3 shows the NMapWin Scan tab, where you can select these options. If you're a command-line fan, you see the command-line parameters displayed in the lower-left corner of the NMapWin screen. This helps when you know what you want to do and the command-line help isn't enough.

If you connect to a single port (as opposed to several all at one time) without making too much noise, you might be able to evade your IDS/IPS system. This is a good test of your IDS/IPS and firewall systems, so assess your logs to see what they saw during this process.

NetScanTools Pro

NetScanTools Pro (`www.netscantools.com`) is a nice all-in-one commercial tool for gathering general network information, such as the number of unique IP addresses, NetBIOS names, and MAC addresses. It also has a neat feature that allows you to fingerprint the operating systems of various hosts. Figure 8-4 shows the OS fingerprint results while scanning a Linksys router/firewall.

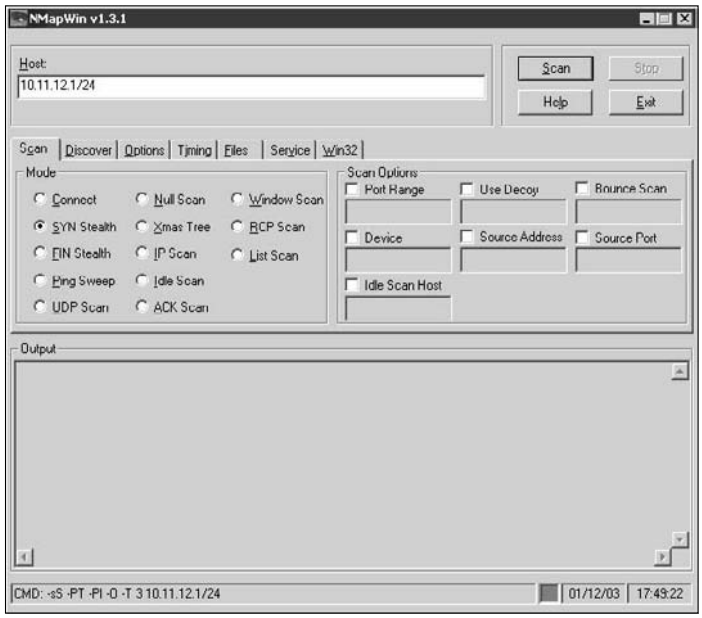


Figure 8-3: In-depth port-scanning options in NMapWin.

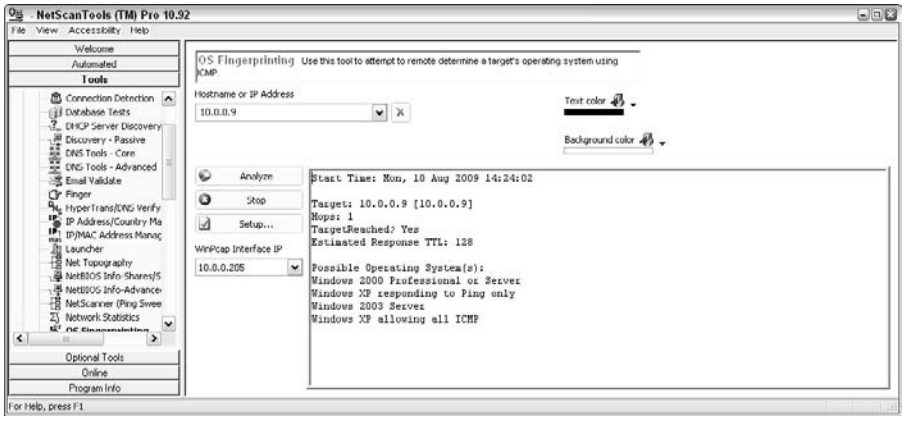


Figure 8-4: NetScan Tools Pro OS finger-printing tool.

Countermeasures against ping sweeping and port scanning

Enable only the traffic you need to access internal hosts — preferably as far as possible from the hosts you're trying to protect — and deny everything else. This goes for standard ports, such as TCP 80 for HTTP and ICMP for ping requests. You apply these rules in two places:

- External router for inbound traffic
- Firewall for outbound traffic

Configure firewalls to look for potentially malicious behavior over time (such as the number of packets received in a certain period of time), and have rules in place to cut off attacks if a certain threshold is reached, such as 10 port scans in one minute or 100 consecutive ping (ICMP) requests.

Most firewalls and IDSes/IPSes can detect such scanning and cut it off in real time.

You *can* break applications on your network when restricting network traffic, so make sure that you analyze what's going on and understand how applications and protocols are working before you disable any type of network traffic.

SNMP scanning

Simple Network Management Protocol (SNMP) is built in to virtually every network device. Network management programs (such as HP OpenView and LANDesk) use SNMP for remote network host management. Unfortunately, SNMP also presents security vulnerabilities.

Vulnerabilities

The problem is that most network hosts run SNMP enabled with the default read/write community strings of public/private. The majority of network devices I come across have SNMP enabled and don't even need it!

If SNMP is compromised, a hacker can gather such network information as ARP tables, usernames, and TCP connections to attack your systems further. If SNMP shows up in port scans, you can bet that a malicious attacker will try to compromise the system. Figure 8-5 shows how GFI LANguard determined the NetWare version running (Version 6, Service Pack 3) by simply querying a host running unprotected SNMP.

Here are some other utilities for SNMP enumeration:

- The commercial tools NetScanTools Pro and Essential NetTools
- Free Windows GUI-based Getif
- Free Windows text-based SNMPUTIL (`www.wtcs.org/snmp4tpc/FILES/Tools/SNMPUTIL/SNMPUTIL.zip`)

You can use Getif to enumerate systems with SNMP enabled, as shown in Figure 8-6.

In this test, I was able to glean a lot of information from a wireless access point, including model number, firmware revision, and system uptime. All this could be used against the host if an attacker wanted to exploit a known

vulnerability in this particular system. By digging in further, I was able to discover several management interface usernames on this access point, as shown in Figure 8-7. You certainly don't want to show the world this information.

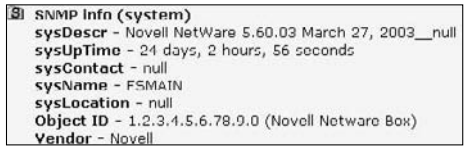


Figure 8-5: Information gathered by querying a vulnerable SNMP host.

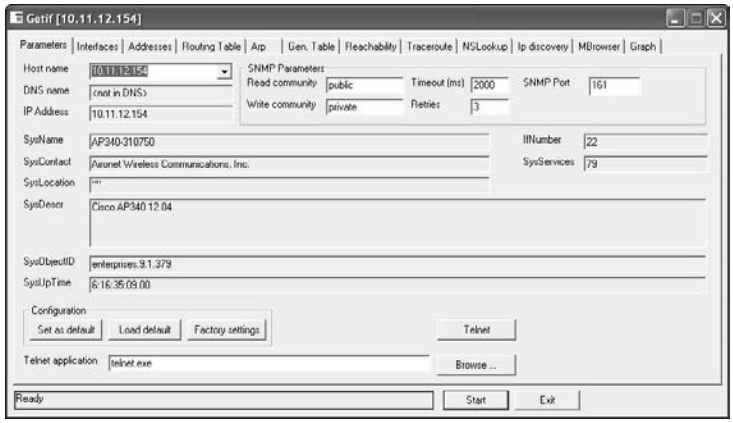


Figure 8-6: General SNMP information gathered by Getif.

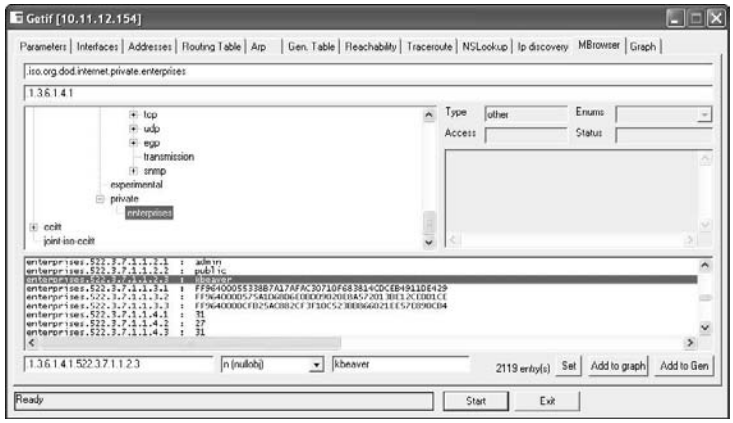
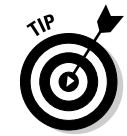


Figure 8-7: Management interface user IDs gleaned via Getif's SNMP browsing function.



For a list of vendors and products affected by the well-known SNMP vulnerabilities, refer to `www.cert.org/advisories/CA-2002-03.html`.

Countermeasures against SNMP attacks

Preventing SNMP attacks can be as simple as A-B-C:

- **A**lways disable SNMP on hosts if you're not using it — period.
- **B**lock the SNMP ports (UDP ports 161 and 162) at the network perimeter.
- **C**hange the default SNMP community read string from `public` and the default community write string from `private` to another long and complex value that's virtually impossible to guess.

There's technically a "U" that's part of the solution: upgrade. Upgrading your systems (at least the ones you can) to SNMP version 3 can resolve many of the well-known SNMP security weaknesses.

Banner grabbing

Banners are the welcome screens that divulge software version numbers and other system information on network hosts. This banner information might identify the operating system, the version number, and the specific service packs to give the bad guys a leg up on attacking the network. You can grab banners by using either good old telnet or some of the tools I mention, such as Nmap and SuperScan.

telnet

You can telnet to hosts on the default telnet port (TCP port 23) to see whether you're presented with a login prompt or any other information. Just enter the following line at the command prompt in Windows or UNIX:

```
telnet ip_address
```

You can telnet to other commonly used ports with these commands:

- **SMTP:** `telnet ip_address 25`
- **HTTP:** `telnet ip_address 80`
- **POP3:** `telnet ip_address 110`

Figure 8-8 shows specific version information about an Exchange 2003 server when telnetting to it on port 25. For help with telnet, simply enter `telnet /?` or `telnet help` for specific guidance on using the program.

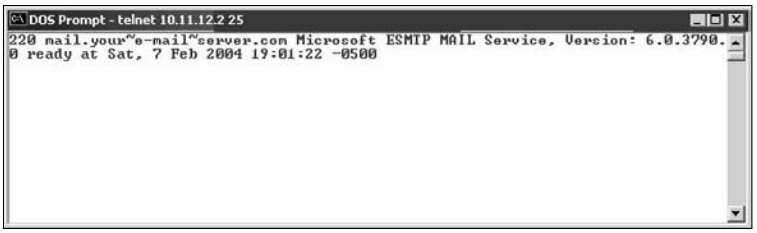


Figure 8-8: Information gathered about Exchange 2003 via telnet.

Countermeasures against banner-grabbing attacks

The following steps can reduce the chance of banner-grabbing attacks:

- If there isn't a business need for services that offer banner information, disable those unused services on the network host.
- If there isn't a business need for the default banners, or if you can customize the banners, configure the network host's application or operating system to either disable the banners or remove information from the banners that could give an attacker a leg up. Check with your specific vendor for information on how to do this.

If you can customize your banners, check with your lawyer about adding a warning banner. It won't stop banner grabbing but will show that the system is private and can help reduce your business liability in the event of a security breach. Here's an example:

> *Warning! This is a private system. All use is monitored and recorded. Any unauthorized use of this system may result in civil and/or criminal prosecution to the fullest extent of the law.*

Firewall rules

As part of your ethical hacking, you can test your firewall rules to make sure they're working as they're supposed to.

Testing

A few tests can verify that your firewall actually does what it says it's doing. You can connect through the firewall on the ports that are open, but what about the ports that can be open but shouldn't be?

Some scanning tools can test for open ports and determine whether traffic is actually allowed to pass through the firewall.

Netcat

Netcat (`http://netcat.sourceforge.net`) can test certain firewall rules without having to test a production system directly. For example, you can check whether the firewall allows port 23 (telnet) through. Follow these steps to see whether a connection can be made through port 23:

1. **Load Netcat on a client machine *inside* the network.**

   This sets up the outbound connection.

2. **Load Netcat on a testing computer *outside* the firewall.**

   This allows you to test from the outside in.

3. **Enter the Netcat listener command on the client (internal) machine with the port number you're testing.**

   For example, if you're testing port 23, enter this command:

   ```
   nc -l -p 23 cmd.exe
   ```

4. **Enter the Netcat command to initiate an inbound session on the testing (external) machine. You must include the following information:**

   - The IP address of the internal machine you're testing
   - The port number you're testing

   For example, if the IP address of the internal (client) machine is 10.11.12.2 and the port is 23, enter this command:

   ```
   nc -v 10.11.12.2 23
   ```

If Netcat presents you with a new command prompt (that's what the `cmd.exe` is for in Step 3) on the external machine, you've connected and can execute commands on the internal machine! This can serve several purposes, including testing firewall rules, network address translation (NAT), port forwarding and — well, uhhhmmm — executing commands on a remote system!

Traffic IQ Pro

A neat commercial tool that specializes in evaluating the performance of packet filtering devices, such as firewalls, is Traffic IQ Pro by Karalon (`www.karalon.com`). With this tool, shown in Figure 8-9, you can connect one network interface card (NIC) on your testing machine to your firewall's internal segment and a second NIC to your firewall's external segment or DMZ and generate generic and/or malicious traffic to see whether your firewall is doing what it's supposed to do. Such a test is great for those annual firewall "rule-based audits" mandated by regulations, such as the Payment Card Industry Data Security Standard (PCI DSS), and by internal audit departments in many organizations.

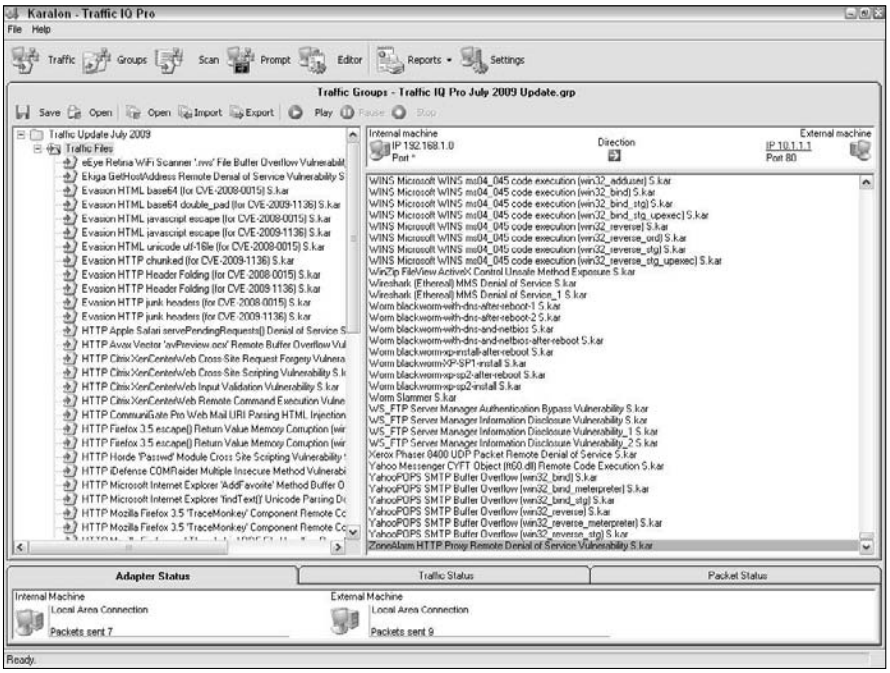


Figure 8-9: Traffic IQ Pro for generating packets and analyzing a firewall's capabilities.

Firewalk

An alternative (and free) firewall rulebase testing tool is Firewalk available via BackTrack (`www.remote-exploit.org/backtrack.html`). Firewalk works by sending TCP and UDP packets with their time-to-live (TTL) incremented by one and, based on the response, determining whether the packets pass through the available ports.

If you're just looking for a basic firewall rulebase audit rather than an in-depth technical analysis, check out nipper (`http://sourceforge.net/projects/nipper`) and Athena FirewallGrader (`www.athenasecurity.net/firewall-grader.html`). They might be just what you need.

Countermeasures against firewall rulebase vulnerabilities

The following countermeasures can prevent a hacker from testing your firewall:

- **Limit traffic to what's needed.**

  Set rules on your firewall (and router, if needed) that passes only traffic that absolutely must pass. For example, have rules in place that allow HTTP inbound traffic to an internal Web server, SMTP inbound traffic to an e-mail server, and HTTP outbound traffic for external Web access.

This is the best defense against someone poking at your firewall.

- **Block ICMP to help prevent an external attacker from poking and prodding your network to see which hosts are alive.**
- **Enable stateful packet inspection on the firewall, if you can. It can block unsolicited requests.**

Network analyzers

A *network analyzer* is a tool that allows you to look into a network and analyze data going across the wire for network optimization, security, and/or troubleshooting purposes. Like a microscope for a lab scientist, a network analyzer is a must-have tool for any security professional.

Network analyzers are often generically referred to as *sniffers,* though that's actually the name and trademark of a specific product from Network Associates, *Sniffer* (the original commercial network analysis tool).

A network analyzer is handy for *sniffing* packets off the wire. A network analyzer is simply software running on a computer with a network card. It works by placing the network card in *promiscuous mode,* which enables the card to see all the traffic on the network, even traffic not destined for the network analyzer's host. The network analyzer performs the following functions:

- Captures all network traffic
- Interprets or decodes what is found into a human-readable format
- Displays the content in chronological order

When assessing security and responding to security incidents, a network analyzer can help you

- View anomalous network traffic and even track down an intruder.
- Develop a baseline of network activity and performance, such as protocols in use, usage trends, and MAC addresses, before a security incident occurs.

When your network behaves erratically, a network analyzer can help you

- Track and isolate malicious network usage
- Detect malicious Trojan-horse applications
- Monitor and track down DoS attacks

Network analyzer programs

You can use one of the following programs for network analysis:

- **WildPackets' OmniPeek** (`www.wildpackets.com/products/distributed_network_analysis/omnipeek_network_analyzer`) is my favorite network analyzer. It does everything I need and more and is very simple to use. EtherPeek is available for Windows operating systems.
- **TamoSoft's CommView** (`www.tamos.com/products/commview`) is a low-cost, Windows-based alternative.
- **Cain & Abel** (`www.oxid.it/cain.html`) is a free multi-functional password recovery tool for performing ARP poisoning, capturing packets, cracking passwords, and more.
- **Wireshark** (`www.wireshark.org`), formerly known as Ethereal, is a free alternative. I download and use this tool if I need a quick fix and don't have my laptop nearby. It's not as user-friendly as most of the commercial products, but it is very powerful if you're willing to learn its ins and outs. Wireshark is available for both Windows and OS X.
- **ettercap** (`http://ettercap.sourceforge.net`) is another powerful (and free) utility for performing network analysis and much more on Windows, Linux, and other operating systems.

Here are a few caveats for using a network analyzer:

- To capture all traffic, you must connect the analyzer to one of the following:
  - A hub on the network
  - A monitor/span/mirror port on a switch
  - A switch that you've performed an ARP poisoning attack on
- If you want to see traffic similar to what a network-based IDS sees you should connect the network analyzer to a hub or switch monitor port on the outside of the firewall, as shown in Figure 8-10, as part of your testing so you can view
  - What's entering your network *before* the firewall filters eliminate the junk traffic.
  - What's leaving your network *after* the traffic passes through the firewall.

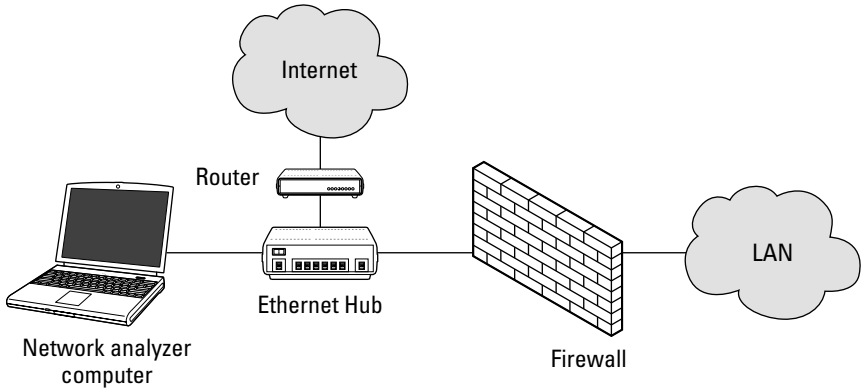


Figure 8-10: Connecting a network analyzer outside the firewall.

Whether you connect your network analyzer inside or outside your firewall, you see immediate results. It can be an overwhelming amount of information, but you can look for these issues first:

- **Odd traffic,** such as:
  - An unusual amount of ICMP packets
  - Excessive amounts of multicast or broadcast traffic
  - Packet types that don't belong, such as NetBIOS in a NetWare environment
- **Internet usage habits,** which can help point out malicious behavior of a rogue insider or system that has been compromised, such as:
  - Web surfing
  - E-mail
  - Instant messaging and other P2P software
- **Questionable usage,** such as:
  - Many lost or oversized packets, indicating hacking tools or malware is present
  - High bandwidth consumption that might point to a Web or FTP server that doesn't belong
- **Reconnaissance probes and system profiling from port scanners and vulnerability assessment tools,** such as a significant amount of inbound traffic from unknown hosts — especially over ports that aren't used very much, such as FTP or telnet.
- **Hacking in progress,** such as tons of inbound UDP or ICMP echo requests, SYN floods, or excessive broadcasts.

- **Nonstandard hostnames on your network.** For example, if your systems are named Computer1, Computer2, and so on, a computer named GEEKz4evUR should raise a red flag.
- **Hidden servers** (especially Web, SMTP, FTP, DNS, and DHCP) that might be eating network bandwidth, serving illegal software, or accessing our network hosts.
- **Attacks on specific applications** that show such commands as `/bin/rm`, `/bin/ls`, `echo`, and `cmd.exe` as well as SQL queries and JavaScript injection, which I cover in Chapter 14.

You might need to let your network analyzer run for quite a while — several hours to several days, depending on what you're looking for. Before getting started, configure your network analyzer to capture and store the most relevant data:

- If your network analyzer permits it, configure it to use a first-in, first-out buffer.

  This overwrites the oldest data when the buffer fills up, but it might be your only option if memory and hard drive space are limited on your network-analysis computer.
- If your network analyzer permits it, record all the traffic into a capture file and save it to the hard drive. This is the ideal scenario — especially if you have a large hard drive, such as 500GB or more.

  You can easily fill a several hundred gigabyte hard drive in a short period. I highly recommend running your network analyzer in what EtherPeek calls *monitor mode*. This allows the analyzer to keep track of what's going on but not capture every single packet. Monitor mode — if supported by your analyzer — is very beneficial and is often all you need.
- When network traffic doesn't look right in a network analyzer, it probably isn't. It's better to be safe than sorry.

  Run a baseline when your network is working normally. When you have a baseline, you can see any obvious abnormalities when an attack occurs.

One thing I like to check for is the "top talkers" on the network. If someone is doing something malicious on the network, such as hosting an FTP server or running Internet file sharing software, using a network analyzer is often the only way you'll find out about it. A network analyzer is also a good tool for detecting systems infected with malware, such as a virus or Trojan horse. Figure 8-11 shows what it looks like to have a suspect protocol or application running on your network.

Looking at your network statistics, such as bytes per second, network utilization, and inbound/outbound packet counts, is also a good way to determine whether something fishy is going on. Figure 8-12 contains network statistics as seen through the powerful CommView network analyzer.

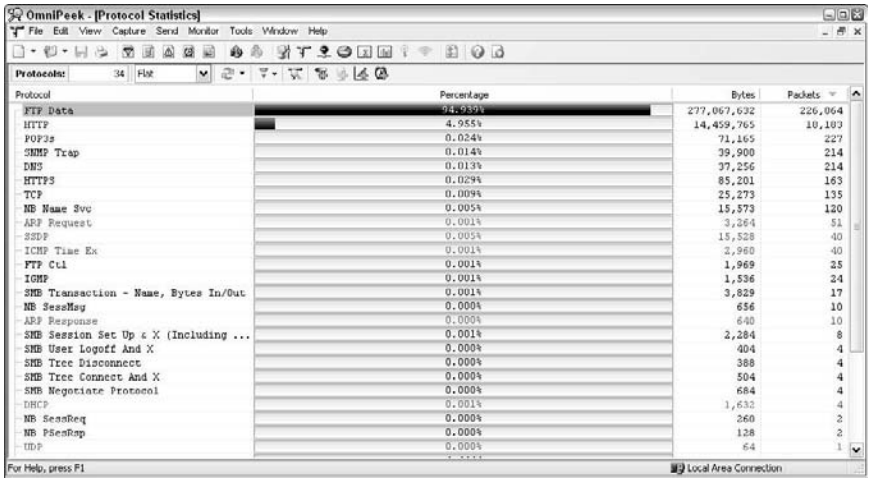


Figure 8-11: OmniPeek can help uncover someone running an illicit system, such as an FTP server.

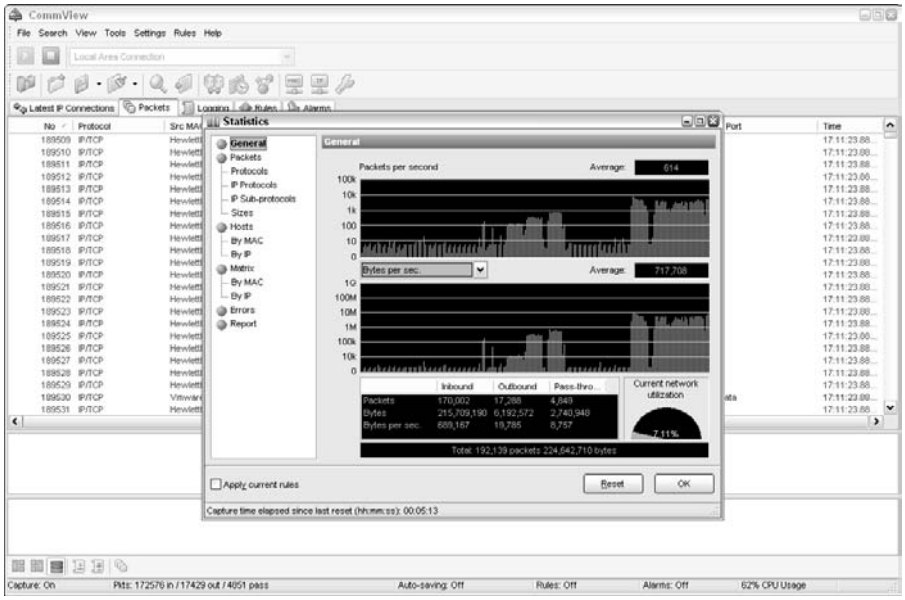


Figure 8-12: Comm View's interface for viewing network statistics.

TamoSoft — the maker of CommView — has another product called NetResident that can track the usage of well-known protocols, such as HTTP, e-mail, FTP, and VoIP. As shown in Figure 8-13, you can use NetResident to monitor Web sessions and play them back.

NetResident also has the ability to perform ARP poisoning, which allows NetResident to see everything on the local network segment. I cover ARP poisoning in the section "The MAC-daddy attack," later in this chapter.

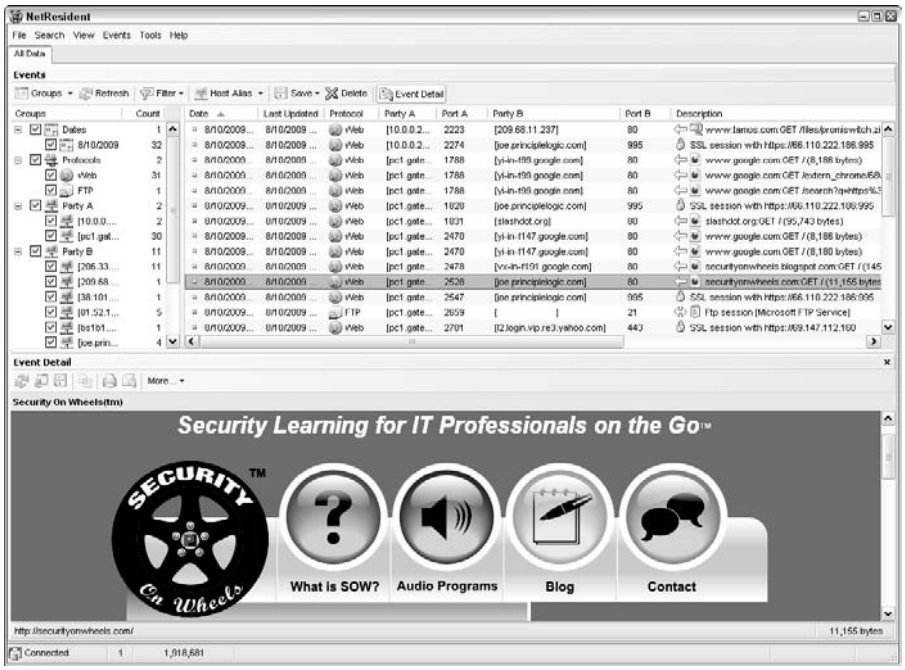


Figure 8-13: NetResident can track Internet usage and ensure security policies are enforced.

Countermeasures against network protocol vulnerabilities

A network analyzer can be used for good or evil. The good is to help ensure your security policies are being followed. The evil is when someone uses a network analyzer against you. A few countermeasures can help prevent someone from using an unauthorized network analyzer, although there's no way to prevent it completely.

If an external attacker or malicious user can connect to your network (physically or wirelessly), he can capture packets on the network, even if you're using an Ethernet switch.

Physical security

Ensure that adequate physical security is in place to prevent someone from plugging into your network:

- **Keep the bad guys out of your server room and wiring closet.** Ensure that the Web, telnet, and SSH management interfaces on your Ethernet switches are especially secure to keep someone from changing the switch port configuration and seeing everything going across the wire.
- **Make sure that unsupervised areas, such as an unoccupied lobby or training room, don't have live network connections.**

Network analyzer detection

You can use a network- or host-based utility to determine whether someone is running an unauthorized network analyzer on your network:

- **Sniffdet** (`http://sniffdet.sourceforge.net`) for UNIX-based systems
- **PromiscDetect** (`http://ntsecurity.nu/toolbox/promiscdetect`) for Windows

These tools enable you to monitor the network for Ethernet cards that are running in promiscuous mode. You simply load the programs on your computer, and the programs alert you if they see promiscuous behaviors on the network (Sniffdet) or local system (PromiscDetect).

The MAC-daddy attack

Attackers can use ARP (Address Resolution Protocol) running on your network to make their systems appear as your system or another authorized host on your network.

ARP spoofing

An excessive number of ARP requests can be a sign of an *ARP spoofing* attack (also called *ARP poisoning*) on your network.

A client running a program, such as dsniff (`www.monkey.org/~dugsong/dsniff`) or Cain & Abel (`www.oxid.it/cain.html`), can change the ARP tables — the tables that store IP addresses to *media access control (MAC)* address mappings — on network hosts. This causes the victim computers to think they need to send traffic to the attacker's computer rather than to the true destination computer when communicating on the network. ARP spoofing is used during man-in-the-middle (MITM) attacks.

Spoofed ARP replies can be sent to a switch, which reverts the switch to *broadcast mode* and essentially turns it into a hub. When this occurs, an attacker can sniff every packet going through the switch and capture anything and everything off the network.

This security vulnerability is inherent in how TCP/IP communications are handled.

Here's a typical ARP spoofing attack with a hacker's computer (Hacky) and two legitimate network users' computers (Joe and Bob):

1. Hacky poisons the ARP caches of victims Joe and Bob by using dsniff, ettercap, or a utility he wrote.

2. Joe associates Hacky's MAC address with Bob's IP address.
3. Bob associates Hacky's MAC address with Joe's IP address.
4. Joe's traffic and Bob's traffic are sent to Hacky's IP address first.
5. Hacky's network analyzer captures Joe's and Bob's traffic.

If Hacky is configured to act like a router and forward packets, it forwards the traffic to its original destination. The original sender and receiver never know the difference!

Using Cain & Abel for ARP poisoning

You can perform ARP poisoning on your switched Ethernet network to test your IDS/IPS or to see how easy it is to turn a switch into a hub and capture anything and everything with a network analyzer.

ARP poisoning can be hazardous to your network's hardware and health, causing downtime and more. So be careful!

Perform the following steps to use Cain & Abel for ARP poisoning:

1. **Load Cain & Abel and then click the Sniffer tab to enter the network analyzer mode.**

   The Hosts page opens by default.

2. **Click the Start/Stop APR icon (the yellow and black circle).**

   The ARP poison routing (how Cain & Abel refers to ARP poisoning) process starts and enables the built-in sniffer.

3. **If prompted, select the network adapter in the window that appears and then click OK.**
4. **Click the blue + icon to add hosts to perform ARP poisoning on.**
5. **In the MAC Address Scanner window that appears, ensure the All Hosts in My Subnet option is selected and then click OK.**
6. **Click the APR tab (the one with the yellow-and-black circle icon) to load the APR page.**
7. **Click the white space under the uppermost Status column heading (just under the Sniffer tab).**

   This re-enables the blue + icon.

8. **Click the blue + icon and the New ARP Poison Routing window shows the hosts discovered in Step 3.**
9. **Select your default route (in my case, 10.11.12.1).**

   The right-hand column fills with all the remaining hosts, as shown in Figure 8-14.

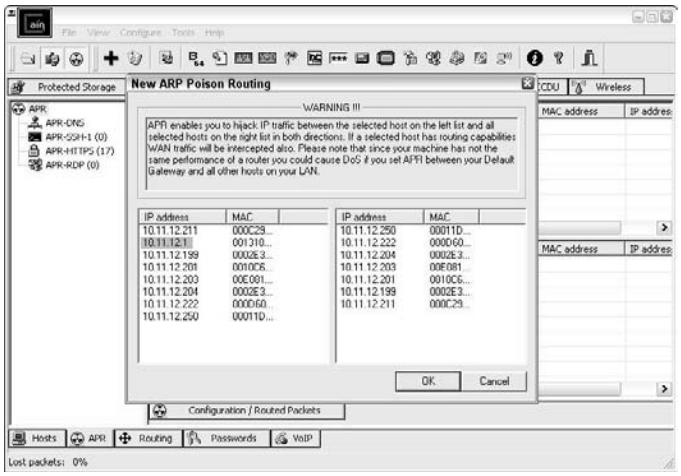


Figure 8-14: Selecting your victim hosts for ARP poisoning in Cain & Abel.

10. **Ctrl+click all the hosts in the right column that you want to poison.**
11. **Click OK and the ARP poisoning process starts.**

    This process can take anywhere from a few seconds to a few minutes depending on your network hardware and each hosts' local TCP/IP stack. The results of ARP poisoning on my test network are shown in Figure 8-15.

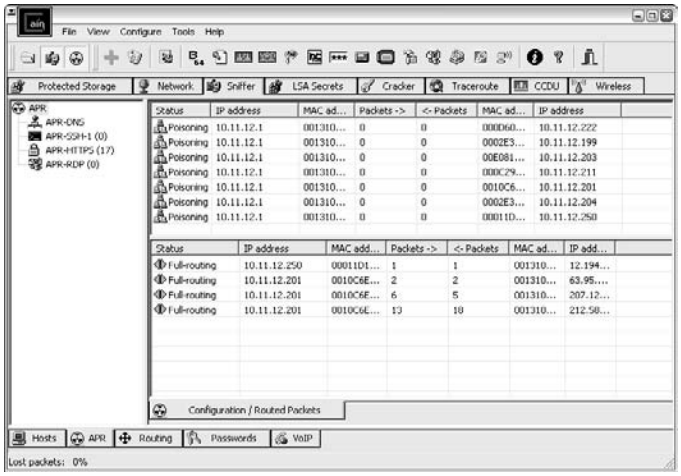


Figure 8-15: ARP poisoning results in Cain & Abel.

12. **You can use Cain & Abel's built-in passwords feature to capture passwords traversing the network to and from various hosts simply by clicking the Passwords tab.**

The preceding steps show how easy it is to exploit a vulnerability and prove that Ethernet switches aren't all they're cracked up to be from a security perspective.

MAC address spoofing

MAC address spoofing tricks the *switch* into thinking your computer is something else. You simply change your computer's MAC address and masquerade as another user.

You can use this trick to test access control systems, such as your IDS/firewall, and even your operating system login controls that check for specific MAC addresses.

UNIX-based systems

In UNIX and Linux, you can spoof MAC addresses with the ifconfig utility. Follow these steps:

1. **While logged in as root, use ifconfig to enter a command that disables the network interface.**

   Insert the network interface number that you want to disable (usually, eth0) into the command, like this:

   ```
   [root@localhost root]# ifconfig eth0 down
   ```

2. **Enter a command for the MAC address you want to use.**

   Insert the fake MAC address and the network interface number (eth0) into the command again, like this:

   ```
   [root@localhost root]# ifconfig eth0 hw ether
   new_mac_address
   ```

You can use a more feature-rich utility called GNU MAC Changer (`www.alobbs.com/macchanger`) for Linux systems.

Windows

You can use regedit to edit the Windows Registry, but I like using a neat Windows utility called SMAC (`www.klcconsulting.net/smac`), which makes MAC spoofing a simple process. Follow these steps to use SMAC:

1. **Load the program.**
2. **Select the adapter for which you want to change the MAC address.**
3. **Enter the new MAC address in the New Spoofed MAC Address fields and click the Update MAC button.**

4. **Stop and restart the network card with these steps:**

   a. *Right-click the network card in Network and Dialup Connections and then choose Disable.*

   b. *Right-click again and then choose Enable for the change to take effect.*

      You might have to reboot for this to work properly.

5. **Click the Refresh button in the SMAC interface.**

To reverse Registry changes with SMAC, follow these steps:

1. **Select the adapter for which you want to change the MAC address.**
2. **Click the Remove MAC button.**
3. **Stop and restart the network card with these steps:**

   a. *Right-click the network card in Network and Dialup Connections and then choose Disable.*

   b. *Right-click again and then choose Enable for the change to take effect.*

      You might have to reboot for this to work properly.

4. **Click the Refresh button in the SMAC interface.**

   You should see your original MAC address again.

Countermeasures against ARP poisoning and MAC address spoofing attacks

A few countermeasures on your network can minimize the effects of an attack against ARP and MAC addresses:

- **Prevention:** You can prevent MAC address spoofing if your switches can enable port security to prevent automatic changes to the MAC address tables.

  No realistic countermeasures for ARP poisoning exist. The only way to prevent ARP poisoning is to create and maintain static ARP entries in your switches for every host on the network. This is something that hardly any network administrator has time to do in today's rat race.

- **Detection:** You can detect these two types of hacks through an IDS, IPS, or a standalone MAC address monitoring utility.

  Arpwatch (`http://linux.maruhn.com/sec/arpwatch.html`) is a Linux-based program that alerts you via e-mail when it detects changes in MAC addresses associated with specific IP addresses on the network.

Denial of service

Denial of service (DoS) attacks are among the most common hacker attacks. A hacker initiates so many invalid requests to a network host that the host uses all its resources responding to the invalid requests and ignores the legitimate requests.

DoS attacks

DoS attacks against your network and hosts can cause systems to crash, data to be lost, and every user to jump on your case wondering when Internet access will be restored.

Here are some common DoS attacks that target an individual computer or network device:

- **SYN floods:** The attacker floods a host with TCP SYN packets.
- **Ping of Death:** The attacker sends IP packets that exceed the maximum length of 65,535 bytes, which can ultimately crash the TCP/IP stack on many operating systems.
- **WinNuke:** This attack can disable networking on older Windows 95 and Windows NT computers.

Distributed DoS (DDoS) attacks have an exponentially greater impact on their victims. One of the most famous was the DDoS attack against eBay, Yahoo!, CNN, and dozens of other Web sites by a hacker known as MafiaBoy. While updating this book to the third edition, there was a highly publicized DDoS attack against Twitter, Facebook, and other social media sites. The attack was apparently aimed at one user from Georgia (the former Soviet country not the state where I live) but it affected everyone using these sites. I couldn't "tweet" and many of my friends and family members couldn't see what everyone was blabbing about on Facebook (oh, the humanity!). Think about this: When hundreds of millions of people can be taken offline by one targeted DDoS attack, you can see why understanding the dangers of denial of service against your business's systems and applications is important.

DoS and DDoS attacks can be carried out with tools that the attacker either writes or downloads from the Internet. These are good tools to test your network's IDS/IPS and firewalls for denial of service weaknesses. You can find programs that allow actual attacks and programs, such as Karalon's Traffic IQ Pro, that let you send controlled attacks.

Testing

Denial of service testing is one of the most difficult security checks you can run. There just aren't enough of you and your computers to go around. Don't fret, you can run a few tests to see where you're weak. Your first test should be a search for DoS vulnerabilities from a vulnerability-scanning perspective. Using vulnerability scanners, such as QualysGuard (`www.qualys.com`) and WebInspect (`www.spidynamics.com`), you can find missing patches and configuration weaknesses that can lead to denial of service.

During a recent security assessment project, QualysGuard found a vulnerability in an older version OpenSSL running on a Web server. As with most DoS findings, I didn't actually exploit the vulnerability because I didn't want to take down the production system. Instead, I listed it as a "medium priority" vulnerability — an issue that had the potential to be exploited. My client pushed back and said OpenSSL wasn't on the system. With permission, I downloaded the exploit code available on the Internet, compiled it, and ran it against my client's server. Sure enough, it took the server offline.

At first, my client thought it was a fluke, but after taking the server offline again, he bought into the vulnerability. It ended up that he was using an OpenSSL derivative, hence the vulnerability. Had my client not fixed the problem, there could have been any number of attackers around the world taking — and keeping — this production system offline, which could have been both tricky and time consuming to troubleshoot. Not good for business!

Don't test for DoS unless you have test systems or can perform controlled tests with the proper tools. Poorly planned DoS testing is a job search in the making. It's like trying to delete data from a network share and hoping that the access controls in place are going to prevent it.

Other DoS testing tools worth checking out are UDPFlood (`www.foundstone.com/us/resources/proddesc/udpflood.htm`), Blast (`www.foundstone.com/us/resources/proddesc/blast.htm`), NetScanTools Pro, and CommView.

Countermeasures against DoS attacks

Most DoS attacks are difficult to predict, but they can be easy to prevent:

- **Test and apply security patches (including service packs and firmware updates) as soon as possible** for network hosts, such as routers and firewalls, as well as for server and workstation operating systems.
- **Use an IDS or IPS to monitor regularly for DoS attacks.** You can run a network analyzer in *continuous capture* mode if you can't justify the cost of an all-out IDS or IPS solution and use it to monitor for DoS attacks.

- **Configure firewalls and routers to block malformed traffic.** You can do this only if your systems support it, so refer to your administrator's guide for details.
- **Minimize IP spoofing** by filtering out external packets that appear to come from an internal address, the local host (127.0.0.1), or any other private and non-routable address, such as 10.x.x.x, 172.16.x.x–172.31.x.x, or 192.168.x.x.
- **Block all ICMP traffic inbound to your network unless you specifically need it.** Even then, you should allow it to come in only to specific hosts.
- **Disable all unneeded TCP/UDP small services,** such as echo and chargen.

Establish a baseline of your network protocols and traffic patterns before a DoS attack occurs. That way, you know what to look for. And periodically scan for such potential DoS vulnerabilities as rogue DoS software installed on network hosts.

Work with a *minimum necessary* mentality (not to be confused with having too many beers) when configuring your network devices, such as firewalls and routers:

- **Identify traffic that is necessary for approved network usage.**
- **Allow the traffic that's needed.**
- **Deny all other traffic.**

If worse comes to worst, you'll need to work with your ISP and see whether they can block DoS attacks on their end.

Common Router, Switch, and Firewall Weaknesses

In addition to the more technical exploits that I cover in this chapter, some high-level security vulnerabilities commonly found on network devices can create many problems.

Unsecured interfaces

You want to ensure that HTTP and telnet interfaces to your routers, switches, and firewall aren't configured with a blank, default, or otherwise easy-to-guess password. This sounds like a no-brainer but it's one of the

most common weaknesses. When a malicious insider or other attacker gains access to your network devices, he owns the network. He can then lock out administrative access, setup backdoor user accounts, reconfigure ports, and even bring down the entire network without you ever knowing.

I once found a simple password that a systems integrator had configured on a Cisco ASA firewall and was able to login to the firewall with full administrative rights. Just imagine what could happen in this situation if someone with malicious intent came across this password. Lesson learned: Know what your vendors are doing and keep an eye on them!

Another weakness is related to HTTP and telnet being enabled and used on many network devices. Care to guess why this is a problem? Well, anyone with some free tools and a few minutes of time can sniff the network and capture login credentials for these systems when they're being sent in cleartext. When that happens, anything goes.

IKE weaknesses

Businesses running a VPN on a router or firewall are common. If you fall into this category, chances are good that your VPN is running the Internet Key Exchange (IKE) protocol, which has a couple of well-known exploitable weaknesses.

First, it's possible to crack IKE "aggressive mode" pre-shared keys using Cain & Abel and the IKECrack tool (`http://ikecrack.sourceforge.net`). Second, some IKE configurations, such as those in certain Cisco PIX firewalls, can be taken offline. All the attacker has to do is send 10 packets per second at 122 bytes each and you have a DoS attack on your hands.

You can manually poke around to see whether your router, switches, and firewalls are vulnerable to these issues, but the best way to find this information is to use a well-known vulnerability scanner, such as QualysGuard (`www.qualys.com`). After you find which vulnerabilities exist, you can take things a step further by using the Cisco Global Exploiter tool (available via the BackTrack toolset). To run Cisco Global Exploiter, download and burn the BackTrack ISO image to CD or boot the image directly through VMWare or VirtualBox. After you enter the BackTrack GUI, click Backtrack, then Vulnerability Identification, then Cisco Global Exploiter, and enter the command `perl cge.pl ip_address exploit_number`, as shown in Figure 8-16.

Good scanners and exploitation tools will save you a ton of time and effort that you can spend on other, more important things, such as Facebook and Twitter.

Figure 8-16: Cisco Global Exploiter tool for exploiting well-known Cisco weaknesses.

General Network Defenses

Regardless of the specific attacks against your system, a few good practices can help prevent many network problems:

- **Use stateful inspection rules that monitor traffic sessions for firewalls.** This can help ensure that all traffic traversing the firewall is legitimate and can prevent DoS attacks and other spoofing attacks.
- **Implement rules to perform packet filtering** based on traffic type, TCP/UDP ports, IP addresses, and even specific interfaces on your routers before the traffic is allowed to enter your network.
- **Use proxy filtering and Network Address Translation (NAT).**
- **Find and eliminate fragmented packets entering your network** (from Fraggle or another type of attack) via an IDS or IPS.
- **Include your network devices in your vulnerability scans.**
- **Ensure your network devices have the latest vendor firmware and patches applied.**
- **Set strong passwords — better yet, passphrases — on all network systems.** I cover passwords in more detail in Chapter 7.
- **Don't use IKE aggressive mode pre-shared keys for your VPN.** If you must, ensure the passphrase is strong and changed periodically (such as every 6–12 months).

- **Always use SSL (HTTPS) or SSH when connecting to network devices.**
- **Segment the network and use a firewall on the following:**
  - The DMZ
  - The internal network
  - Critical subnetworks broken down by business function or department, such as accounting, finance, HR, and research

Chapter 9

Wireless LANs

In This Chapter

- Understanding risks of wireless LANs
- Selecting wireless LAN hacking tools
- Hacking against wireless LANs
- Minimizing wireless network security risks

Wireless local area networks (WLANs, also called Wi-Fi) — specifically, the ones based on the IEEE 802.11 standard — are increasingly being deployed into both business and home networks. Next to Voice over IP (VoIP) and digital video recorders, WLANs are the neatest technology I've used in quite a while. Of course, with any new computing technology comes security issues, and WLANs are no exception. In fact, 802.11 wireless has been the poster child for weak security and network hack attacks for several years running. The stigma of unsecure WLANs is starting to wane but this isn't the time to lower your defenses.

WLANs offer a ton of business value, from convenience to reduced network deployment time. Whether or not your organization allows wireless network access, you probably have it, so testing for WLAN security vulnerabilities is critical. In this chapter, I cover some common wireless network security vulnerabilities that you should test for, and I discuss some cheap and easy countermeasures you can implement to help ensure that WLANs are not more of a risk to your organization than they're worth.

Understanding the Implications of Wireless Network Vulnerabilities

WLANs are very susceptible to attack — even more so than wired networks (discussed in Chapter 8). They have vulnerabilities that can allow an attacker to bring your network to its knees or allow your sensitive information to be extracted right out of thin air. If your WLAN is compromised, you can experience the following problems:

- Loss of network access, including e-mail, Web, and other services that can cause business downtime
- Loss of sensitive information, including passwords, customer data, intellectual property, and more
- Regulatory consequences and legal liabilities associated with unauthorized users gaining access to your business systems

Most of the wireless vulnerabilities are in the 802.11 protocol and how it works. Wireless *access points* (APs) and client systems have some vulnerabilities as well.

For a database of wireless-specific security vulnerabilities, refer to the Wireless Vulnerabilities and Exploits site at `www.wvew.org`. It's sort of a Common Vulnerabilities and Exposures database for the wireless world.

Various fixes have come along in recent years to address these vulnerabilities, but most of these fixes have not been properly applied or are not enabled by default. Your employees might also install rogue WLAN equipment on your network without your knowledge; this is arguably the most serious threat to your wireless security and a pretty difficult one to fight. Even when WLANs are hardened and all the latest patches have been applied, you still might have some serious security problems, such as DoS, man-in-the-middle attacks, and encryption key weaknesses (like you have on wired networks — see Chapter 8), that will likely be around for a while.

A case study with Joshua Wright on hacking wireless networks

Joshua Wright shared with me an interesting story about wireless penetration testing and why the little things always seem to get you.

The Situation

Mr. Wright was onsite for a wireless penetration test for a customer who needed validation on his network design and implementation. The customer had carefully designed the network to provide access to three groups of users: employees, legacy handheld wireless scanners, and guests. Employees were granted access to internal systems and applications but were required to first authenticate to the wireless network using two-factor devices. The legacy handheld wireless scanners were only allowed to access a limited number of needed resources using WPA with pre-shared key authentication. The guest users were restricted to Internet access only over an open wireless network. Mr. Wright's job was to break in to the network and to demonstrate the weaknesses to the customer.

The Outcome

The employee and legacy wireless networks were both using AES-CCMP encryption so there was little chance of getting in that way. Mr. Wright attempted to compromise the pre-shared key used on the legacy network but was unsuccessful after exhausting a dictionary list of common passwords. The employee wireless clients were configured to reject networks without the proper SSID and authentication settings, defeating his attempts to impersonate a legitimate AP. A traceroute on the guest network revealed that it was physically separate from the company WAN.

Mr. Wright was starting to run out of options when he remembered the teaching of spiritual guru Ram Dass who once said, "The quieter you become the more you can hear." Instead of aggressively attempting to exploit the network, Mr. Wright started watching network activity on the guest network with tcpdump, thinking that perhaps he'd find an employee system that was misconfigured and on the wrong network.

After starting tcpdump, Mr. Wright started seeing broadcast and multicast traffic from source IP addresses that didn't belong in the DHCP pool for the guest network. The sources Mr. Wright was seeing were not from guest systems at all, but rather belonged to devices on the employee and legacy device networks. While still connected to the guest network, Mr. Wright manually configured his adapter with an unused IP address from the employee network, which granted him unrestricted access to internal systems, including an unpatched Windows 2003 server that was vulnerable to the RPC DCOM interface overflow exploit.

Later discussion with the customer revealed that the company WAN connection was deemed too slow for downloading large patch updates, so administrators would temporarily connect internal systems to the guest network to download the patches and disconnect. One forgotten system was configured to bridge multiple interfaces, granting access to the internal networks from the guest network. By simply listening to what the network was trying to tell him, Mr. Wright was able to bypass the well-planned intentions for security.

Joshua Wright is a senior security analyst for InGuardians, Inc., a computer security consulting services organization, and a senior instructor for the SANS Institute. Joshua specializes in attacking wireless systems, and he's published books, papers, and countless tools on his Web site, `www.willhackforsushi.com`. When he's not hacking wireless networks, Joshua seeks any opportunity to void the warranty on electronic devices.

Choosing Your Tools

Several great WLAN security tools are available for both the Windows and UNIX platforms. The UNIX tools — which run mostly on Linux and BSD — can be a bear to configure and run properly if the planets and stars are not properly aligned but they're worth it if you can endure the pain. The PC Card services in Linux are the trickiest to set up, depending on your type of WLAN card and your Linux version.

Don't get me wrong — the UNIX-based tools are excellent at what they do. Programs such as Kismet (`www.kismetwireless.net`) and Wellenreiter (`http://sourceforge.net/projects/wellenreiter/`) offer many features that most Windows-based applications don't have. These programs run really well if you have all the Linux dependencies installed. They also offer many features that you don't need when assessing the security of your WLAN.

If you want the power of the security tools that run on Linux, but you're not interested in installing and learning much about Linux or don't have the time to download and set up many of its popular security tools, I highly recommend you check out BackTrack (`www.remote-exploit.org/backtrack.html`). The bootable Slackware Linux-based CD "automagically" detects your hardware settings and comes with a slew of security tools that are relatively easy to use. Alternative bootable (or "live") CDs include the Fedora Linux-based Network Security Toolkit (`www.networksecuritytoolkit.org`) and the Knoppix Linux-based Security Tools Distribution (`http://s-t-d.org`). A complete listing of live bootable Linux toolkits is at `www.livecdlist.com`.

Having said this about UNIX-based tools, the good thing is that in the past couple of years, Windows-based tools have greatly improved — especially the commercial tools.

Most of the tests I outline in this chapter require only Windows-based utilities. My favorite tools for assessing wireless networks in Windows are as follows:

- NetStumbler (`www.netstumbler.com`)
- AirMagnet (now Fluke) WiFi Analyzer (`http://www.airmagnet.com/products/wifi_analyzer`)
- WildPackets' OmniPeek (`www.wildpackets.com/products/distributed_network_analysis/omnipeek_network_analyzer`)
- Elcomsoft Wireless Security Auditor (`www.elcomsoft.com/ewsa.html`)
- aircrack (`http://aircrack-ng.org`)

You also need the proper hardware. A good setup I use is a laptop PC with an Orinoco 802.11b PC Card (formerly made by Lucent, now Proxim). This card is not only compatible with NetStumbler, but it also allows you to connect

an external antenna. Another bonus is that most wireless security tools are very friendly with the Orinoco card. A lot of security tool support is available for the Prism2 chipset found in wireless cards by Belkin, D-Link, Linksys, and more. I've also get good results using AirMagnet's WiFi Analyzer with a Netgear WAG511 v2 or Linksys WPC55AG card.

Before you purchase a wireless PC Card or PCI adapter, verify what chipset it has to ensure compatibility with the majority of security tools. The Seattle Wireless Hardware Comparison page (`www.seattlewireless.net/index.cgi/HardwareComparison`) is a good reference for this type of information. Also, be sure to refer to the hardware requirements list from your commercial wireless tool vendors and any README files that come with free tools.

You can also use a handheld wireless security testing device, such as the handy Digital Hotspotter by Canary Wireless (`www.canarywireless.com`) or the ultra-powerful AirMagnet Handheld Analyzer (`www.airmagnet.com/products/handheld_analyzer`). The former is great for rooting out rogue wireless devices, and the latter is an all-out network analyzer that's great for testing various security settings on your WLAN.

An external antenna is also something to consider as part of your arsenal. I have had good luck running tests without an antenna, but your mileage may vary. If you're performing a walkthrough of your facilities to test for wireless signals, for example, using an additional antenna increases your odds of finding both legitimate and (more important) unauthorized wireless systems. You can choose among three types of wireless antennas:

- **Omnidirectional:** Transmits and receives wireless signals in 360 degrees over shorter distances, such as in boardrooms or reception areas. These antennas, also known as dipoles, typically come installed on APs from the factory.
- **Semidirectional:** Transmits and receives directionally focused wireless signals over medium distances, such as down corridors and across one side of an office or building.
- **Directional:** Transmits and receives highly focused wireless signals over long distances, such as between buildings. This antenna, also known as a high-gain antenna, is the antenna of choice for wireless hackers driving around cities looking for vulnerable APs — an act known as *wardriving*.

As an alternative to the antennas described in the preceding list, you can use a nifty can design — called a *cantenna* — made from a Pringles, coffee, or pork-and-beans can. If you're interested in trying this, check out the article at `www.turnpoint.net/wireless/has.html` for details. A simple Internet search turns up a lot of information on this subject, if you're interested. One site in particular (`www.cantenna.com`) sells the Super Cantenna kit — which has worked well for me — for only $49.95. Another good site for cantenna kits is Hugh Pepper's site: `http://mywebpages.comcast.net/hughpep`.

Wireless LAN Discovery

After you have a wireless card and wireless testing software, you're ready to roll. The first tests you should perform gather information about your WLAN, as described in the following sections.

Checking for worldwide recognition

The first test requires only the MAC address of your AP and access to the Internet. You're testing to see whether someone has discovered your WLAN and posted information about it for the world to see. If you're not sure what your AP's MAC address is, you should be able to view it by using the `arp -a` command at a Windows command prompt. You might have to ping the access point's IP address first so the MAC address is loaded into your ARP cache. Figure 9-1 shows what this can look like.

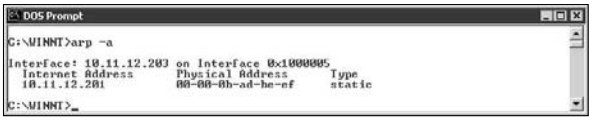


Figure 9-1: Finding the MAC address of an AP by using arp.

After you have the AP's MAC address, browse to the WiGLE database of WLANs (`www.wigle.net`) to see whether your AP is listed. You have to register with the site to perform a database query, but it's worth it. After you select the Query link and log in, you see a screen similar to Figure 9-2. You can enter such AP information as geographical coordinates, but the simplest thing to do is enter your MAC address in the format shown in the example for the BSSID or MAC text box.

If your AP is listed, someone has discovered it — most likely via wardriving — and has posted the information for others to see. You need to start implementing the security countermeasures listed in this chapter as soon as possible to keep others from using this information against you! You can check other WLAN lookup sites, such as `www.wifimaps.com` and `www.wifinder.com`, to see whether your AP is listed there as well.

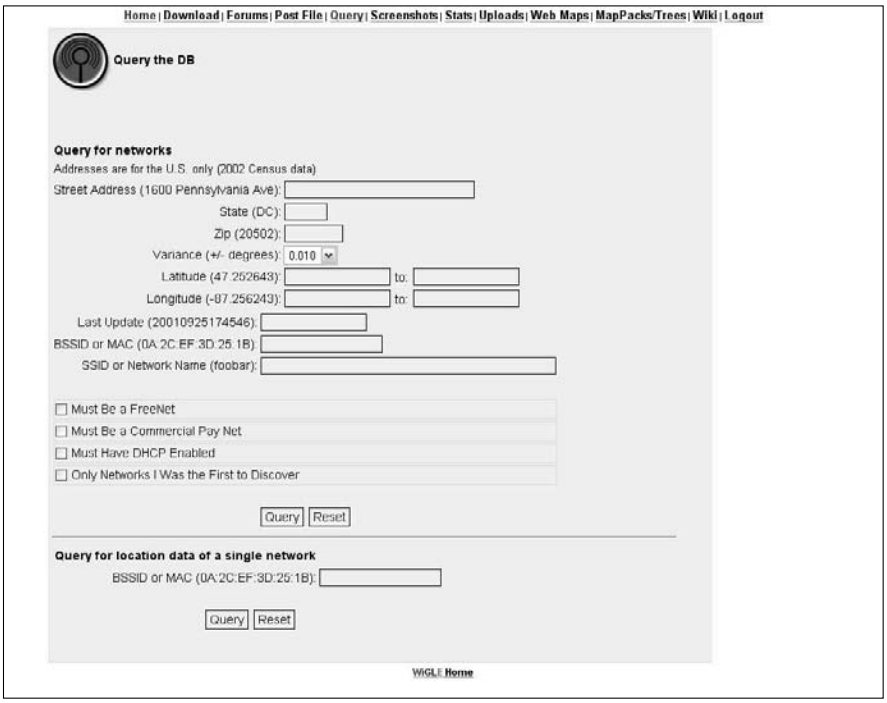


Figure 9-2: Searching for your wireless APs using the WiGLE database.

Scanning your local airwaves

Monitor the airwaves around your building to see what authorized and unauthorized APs you can find. You're looking for the SSID (service set identifier), which is your wireless network name. If you have multiple and separate wireless networks, each one has a unique SSID associated with it.

Here's where NetStumbler comes into play. NetStumbler can discover SSIDs and other detailed information about wireless APs, including the following:

- MAC address
- Name
- Radio channel in use
- Vendor name
- Whether encryption is on or off
- RF signal strength (signal-to-noise ratio)

Figure 9-3 shows an example of what you might see when running NetStumbler in your environment. The information that you see here is what others can see as long as they're in range of your AP's radio signals. NetStumbler and most other tools work by sending a probe-request signal from the client. Any APs within signal range must respond to the request with their SSIDs — that is, if they're configured to broadcast their SSIDs upon request.

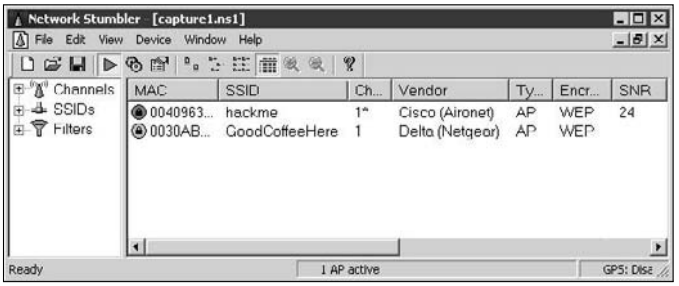


Figure 9-3: NetStumbler displays detailed data on APs.

When you're using certain wireless security assessment tools, including NetStumbler and AirMagnet WiFi Analyzer, your adapter might enter passive monitoring mode. This means you can no longer communicate with other wireless hosts or APs while the program is loaded. Also, some programs require a specialized driver for your wireless card that often disables normal WLAN functionality. If this is the case, you need to roll back (reinstall) the original adapter's driver (supplied by the vendor) to restore the standard functions of your adapter when you complete your testing.

Wireless Network Attacks and Countermeasures

Various malicious hacks — including DoS attacks — can be carried out against your WLAN. This includes forcing APs to reveal their SSIDs during the process of being disassociated from the network and rejoining. In addition, hackers can literally jam the RF signal of an AP — especially in 802.11b and 802.11g systems — and force the wireless clients to reassociate to a rogue AP masquerading as the victim AP.

Hackers can create man-in-the-middle attacks by maliciously using such tools as ESSID-jack and monkey-jack and can flood your network with thousands of packets per second by using such packet-generation tools

as Gspoof and LANforge — enough to bring the network to its knees. Even more so than with wired networks, this DoS attack is very difficult to prevent on WLANs.

You can carry out several attacks against your WLAN. The associated countermeasures help protect your network from these vulnerabilities as well as from the malicious attacks previously mentioned. When testing your WLAN security, look out for the following weaknesses:

- Unencrypted wireless traffic
- Weak WEP and WPA pre-shared keys
- Unauthorized APs
- Easily circumvented MAC address controls
- Wireless equipment that's physically accessible
- Default configuration settings

A good starting point for testing is to attempt to attach to your WLAN as an outsider and run a vulnerability assessment tool, such as LANguard. This test enables you to see what others can see on your network, including information on the OS version, open ports on your AP, and even network shares on wireless clients. Figure 9-4 shows the type of information that can be revealed about an AP on your network.

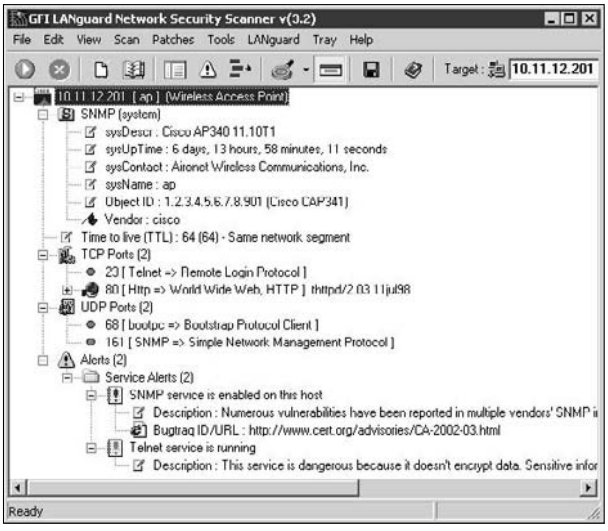


Figure 9-4: A LANguard scan of a potentially vulnerable AP.

Don't overlook Bluetooth

You undoubtedly have various Bluetooth-enabled wireless devices, such as laptops and smartphones, running within your organization. Although vulnerabilities are not as prevalent as they are in 802.11-based Wi-Fi networks, they still exist (currently, over 60 Bluetooth-related weaknesses are listed at `http://nvd.nist.gov`), and quite a few hacking tools take advantage of them. You can even overcome the personal area network distance limitation of Bluetooth's signal (typically just a few meters) and attack Bluetooth devices remotely by building and using a BlueSniper rifle (see the following list for the Web site). Various resources and tools for testing Bluetooth authentication/pairing and data transfer weaknesses include

- **Car Whisperer** (`http://trifinite.org/trifinite_stuff_carwhisperer.html`)
- **Blooover** (`http://trifinite.org/trifinite_stuff_blooover.html`)
- **BlueScanner** (`https://labs.arubanetworks.com`)
- **Bluesnarfer** (`www.alighieri.org/tools/bluesnarfer.tar.gz`)
- **BlueSniper rifle** (`www.tomsguide.com/us/how-to-bluesniper-pt1,review-408.html`)
- **Bluejacking community site** (`www.bluejackq.com`)
- **BTScanner for XP** (`www.pentest.co.uk/src/btscanner_1_0_0.zip`)
- **Smurf** (`www.gatefold.co.uk/smurf`)
- **Detailed presentation on the various Bluetooth attacks** (`http://trifinite.org/Downloads/21c3_Bluetooth_Hacking.pdf`)

Mobile devices have become a completely new dilemma for information security. I honestly think they're one of the greatest risks in any given business. Not only can your mobile devices be hacked via Bluetooth, they also can have serious physical security weaknesses, which can allow a malicious person to gain tons of sensitive information from your organization. A good reference guide for locking down your Bluetooth systems is NIST's Special Publication 800-48, which can be found at `http://csrc.nist.gov/publications/nistpubs/800-48/NIST_SP_800-48.pdf`.

Encrypted traffic

Wireless traffic can be captured directly out of the airwaves, making this communications medium susceptible to eavesdropping. Unless the traffic is encrypted, it's sent and received in cleartext just like on a standard wired network. On top of that, the 802.11 encryption protocols, Wired Equivalent Privacy (WEP) and Wi-Fi Protected Access (WPA), have their own weakness that allows attackers to crack the encryption keys and decrypt the captured traffic. This vulnerability has really helped put WLANs on the map — so to speak.

WEP, in a certain sense, actually lives up to its name: It provides privacy equivalent to that of a wired network, and then some. However, it wasn't intended to be cracked so easily. WEP uses a fairly strong symmetric (shared-key) encryption algorithm called RC4. Hackers can observe encrypted wireless traffic and recover the WEP key because of a flaw in how the RC4 initialization vector (IV) is implemented in the protocol. This weakness is because the IV is only 24 bits long, which causes it to repeat every 16.7 million packets — even sooner in many cases, based on the number of wireless clients entering and leaving the network.

Most WEP implementations initialize WLAN hardware with an IV of 0 and increment it by one for each packet sent. This can lead to the IVs reinitializing — starting over at 0 — approximately every five hours. Given this behavior, WLANs that have a small number of clients transmitting a relatively small rate of wireless packets are normally more secure than large WLANs that transmit a lot of wireless data because there's simply not enough wireless traffic being generated.

Using WEPCrack (`http://wepcrack.sourceforge.net`), AirSnort (`http://airsnort.shmoo.com`), or, my favorite, the aircrack suite (`http://aircrack-ng.org`), hackers need to collect only a few hours' up to a few days' (depending on how much wireless traffic is on the network) worth of packets to break the WEP key. Figure 9-5 shows airodump (which is part of the aircrack suite) capturing WEP initialization vectors, and Figure 9-6 shows aircrack at work cracking the WEP key of my test network.

I'm not a Mac user, but I've heard good things about KisMAC (`http://trac.kismac-ng.org`) for cracking WEP keys among other things.

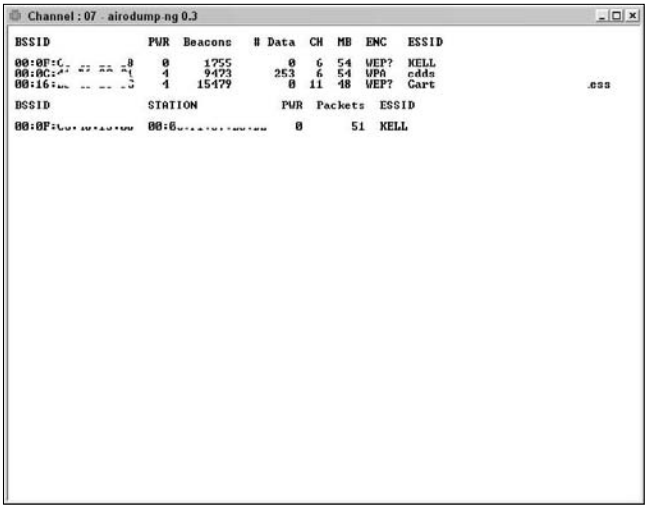


Figure 9-5: Using airodump to capture WEP initialization vectors.

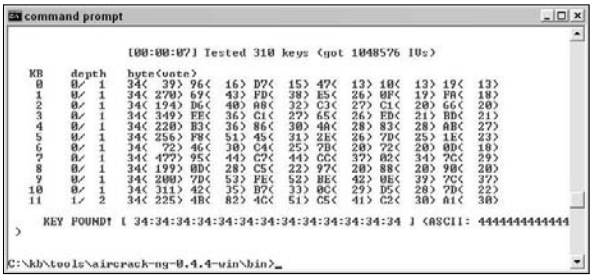


Figure 9-6: Using aircrack to crack WEP.

Airodump and aircrack are very simple to run in Windows. You simply download and extract the aircrack programs, the cygwin Linux simulation environment, and the supporting peek files from the project URL shown earlier and you're ready to capture packets and crack away!

A longer key length, such as 128 bits or 192 bits, doesn't make WEP exponentially more difficult to crack. This is because WEP's static key scheduling algorithm requires that only about 20,000 or so additional packets be captured to crack a key for every extra bit in the key length.

The wireless industry has come up with a solution to the WEP problem called *Wi-Fi Protected Access* (WPA). WPA uses the *Temporal Key Integrity Protocol* (TKIP) encryption system, which fixes all the known WEP issues. WPA2 which replaced the original WPA uses an even stronger encryption method called Counter Mode with Cipher Block Chaining Message Authentication Code Protocol (say that fast three times) — or CCMP — based on the Advanced Encryption Standard (AES). WPA and WPA2 running in "enterprise mode" require an 802.1*x* authentication server, such as a RADIUS server, to manage user accounts for the WLAN. Check with your vendor for WPA updates.

You can also use aircrack to crack WPA and WPA2 pre-shared keys (PSKs). To crack WPA-PSK encryption, you have to wait for a wireless client to authenticate with its access point. A quick (and dirty) way to force the re-authentication process is to send a de-authenticate packet to the broadcast address. This is something my co-author, Peter T. Davis, and I cover in detail in our book, *Hacking Wireless Networks For Dummies*.

You can use airodump to capture packets and then start aircrack (you can also run them simultaneously) to initiate cracking the pre-shared key by using the following command-line options:

```
#aircrack-ng -a2 -w path_to_wordlist <capture file(s)>
```

Another, relatively new, tool you can use for cracking WPA and WPA2 keys is the commercial product Elcomsoft Wireless Security Auditor (EWSA). To use EWSA, you simply capture wireless packets in the tcpdump format (every

WLAN analyzer supports this format), load the capture file into the program, and shortly thereafter you have the PSK. EWSA is a little different because it can crack WPA and WPA2 PSKs in a fraction of the time it would normally take, but there's a caveat. You have to have a computer with a supported NVIDIA or ATI video card. Yep, EWSA doesn't just use the processing power of your CPU — it also harnesses the power and mammoth acceleration capabilities of the video card's graphical processor unit (GPU). Now that's innovation!

The main EWSA interface is shown in Figure 9-7.

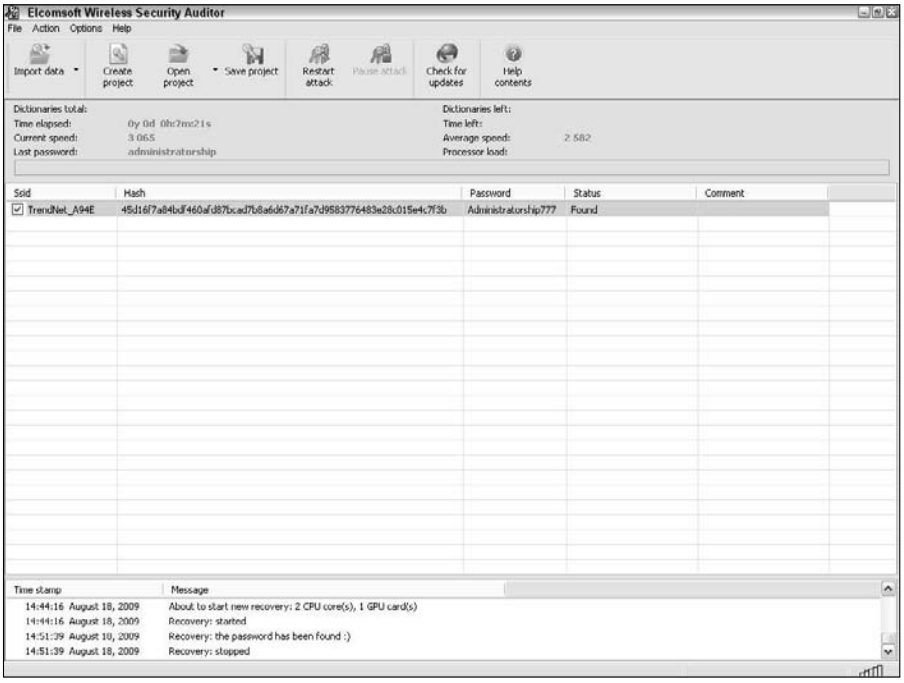


Figure 9-7: Using Elcomsoft Wireless Security Auditor to crack WPA pre-shared keys.

Using EWSA, you can try to crack your WPA/WPA2 PSKs at a rate of up to 50,000 WPA/WPA2 pre-shared keys per second. Compare that to the lowly few hundred keys per second using just the CPU and you can see the value in a tool like this. I always say you get what you pay for.

If you need to use your WLAN analyzer to view traffic as part of your security assessment, you won't see any traffic if WEP is enabled unless you know the WEP key associated with the network. You can enter the key into your analyzer, but just remember that hackers can do the same thing if they're able to crack your WEP key by using one of the tools I mention earlier.

Figure 9-8 shows an example of how you can view protocols on your WLAN by entering the WPA key into OmniPeek via the Capture Options window before you start your packet capture.

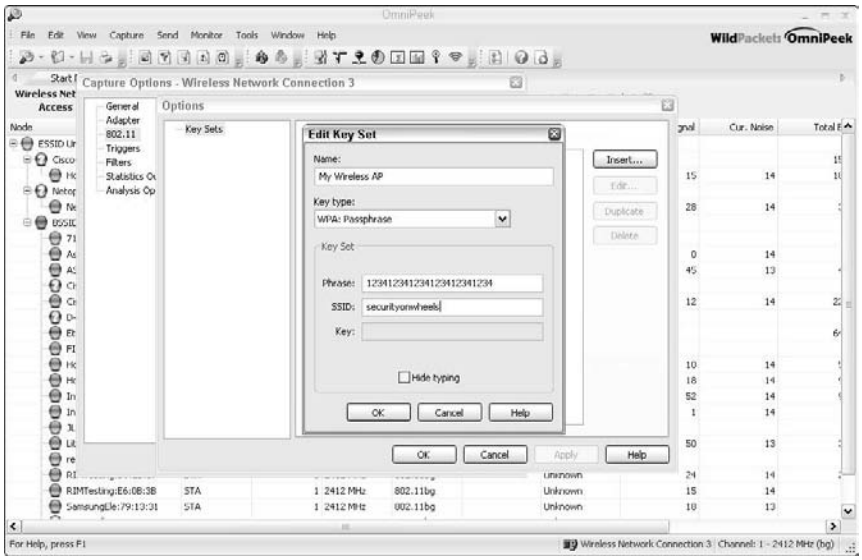


Figure 9-8: Using OmniPeek to view encrypted wireless traffic.

Countermeasures against encrypted traffic attacks

The simplest solution to the WEP problem is to migrate to WPA, or ideally, WPA2, for all wireless communications. You can also use a VPN in a Windows environment — free — by enabling Point-to-Point Tunneling Protocol (PPTP) for client communications. You can also use the IPSec support built into Windows, as well as Secure Shell (SSH), Secure Sockets Layer/Transport Layer Security (SSL/TLS), and other proprietary vendor solutions, to keep your traffic secure. Just keep in mind that there are cracking programs for PPTP, IPSec, and other VPN protocols as well, but overall, you're pretty safe.

Newer 802.11-based solutions exist as well. If you can configure your wireless hosts to regenerate a new key dynamically after a certain number of packets have been sent, the WEP vulnerability can't be exploited. Many AP vendors have already implemented this fix as a separate configuration option, so check for the latest firmware with features to manage key rotation. For instance, the proprietary Cisco LEAP protocol uses per-user WEP keys that offer a layer of protection if you're running Cisco hardware. Again, be careful because cracking programs exist for LEAP, such as *asleap* (`http://asleap.sourceforge.net`).

The 802.11i standard from the IEEE (also called WPA2) integrates the WPA fixes and more. This standard is an improvement over WPA but is not compatible with older 802.11b hardware because of its implementation of the Advanced Encryption Standard (AES) for encryption.

If you're using WPA with a pre-shared key (which is more than enough for small WLANs), ensure that the key contains at least 20 random characters so it isn't susceptible to the offline dictionary attacks available in such tools as aircrack and Elcomsoft Wireless Security Auditor.

Keep in mind that although WEP and weak WPA pre-shared keys are crackable, it's still much better than no encryption at all. Similar to the effect that home security system signs have on would-be home intruders, a wireless LAN running WEP or weak WPA pre-shared keys is not nearly as attractive to a hacker as one without it. Hackers are likely to move on to easier targets unless they really, really want to get into yours.

Rogue wireless devices

Watch out for unauthorized APs and wireless clients that are attached to your network and running in ad-hoc mode.

Educate your users on safe Wi-Fi usage when they're outside of your office as well. Communicate to them the dangers of connecting to unknown WLANs and remind them on a periodic and consistent basis. Otherwise, their systems can be hacked or become infected with malware and guess whose problem it is as soon as they connect back onto your network.

By using NetStumbler or your client manager software, you can test for APs and ad-hoc (or peer) devices that don't belong on your network. You can also use the network monitoring features in a WLAN analyzer, such as OmniPeek and AirMagnet WiFi Analyzer.

Look for the following rogue AP characteristics:

- **Odd SSIDs,** including the popular default ones *linksys* and *wifi*.
- **Odd AP system names —** that is, the name of the AP if your hardware supports this feature. Not to be confused with the SSID.
- **MAC addresses that don't belong on your network.** Look at the first three bytes of the MAC address (the first six numbers), which specify the vendor name. You can perform a MAC-address vendor lookup at `http://standards.ieee.org/regauth/oui/index.shtml` to find information on APs you're unsure of.
- **Weak radio signals,** which can indicate that an AP has been hidden away or is on the outside of your building.

- **Communications across a different radio channel(s) than what your network communicates on.**
- **Degradation in network throughput for any WLAN client.**

In Figure 9-9, NetStumbler has found two potentially unauthorized APs. The ones that stand out are the two with SSIDs of BI and LarsWorld. Notice how they're running on two different channels, two different speeds, and are made by two different hardware vendors. If you know what's supposed to be running on your wireless network (you do, don't you?), unauthorized systems can really stand out.

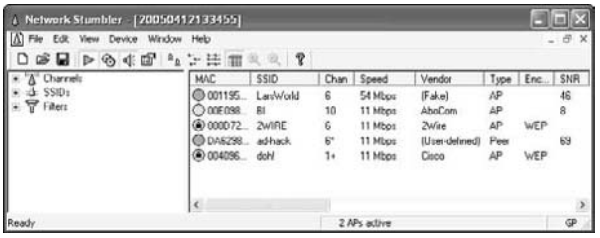


Figure 9-9: NetStumbler showing potentially unauthorized APs.

NetStumbler does have one limitation: It won't find APs that have probe response (SSID broadcast) packets disabled. Kismet — the popular wireless sniffer for Linux and BSD — looks not only for probe responses from APs like NetStumbler does, but also for other 802.11 management packets, such as association responses and beacons. This allows Kismet to detect the presence of "hidden" WLANs.

If the UNIX platform is not your cup of tea, and you're still looking for a quick and dirty way to root out hidden APs, you can create a client-to-AP reconnection scenario that forces the broadcasting of SSIDs using de-authentication packets. You can find detailed instructions in the book I wrote with Peter Davis, *Hacking Wireless Networks For Dummies*.

The safest way to root out hidden APs is to simply search for 802.11 management packets by using a WLAN analyzer, such as AirMagnet Wifi Analyzer or OmniPeek. TamoSoft's CommView for WiFi (`www.tamos.com/products/commwifi`) is also a nice analyzer for this task, and it's very inexpensive to boot.

You can configure OmniPeek to search for 802.11 management packets to root out "hidden" APs by enabling a capture filter on 802.11 management packets, as shown in OmniPeek's options in Figure 9-10.

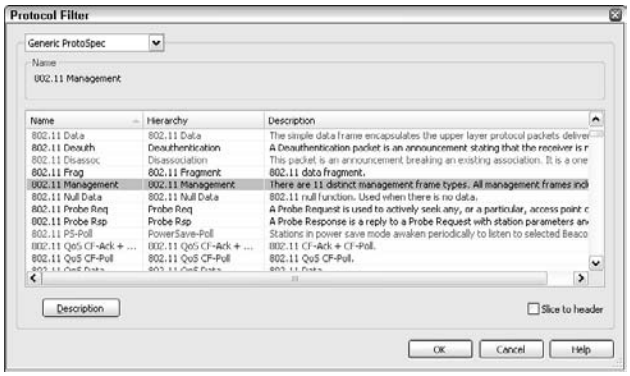


Figure 9-10: OmniPeek can be configured to detect APs that don't broadcast their SSIDs.

Figure 9-11 shows how you can use CommView for WiFi to spot an odd network host; for instance, the Hon Hai and Netgear systems if you know you only use Cisco and Netopia hardware on your network.

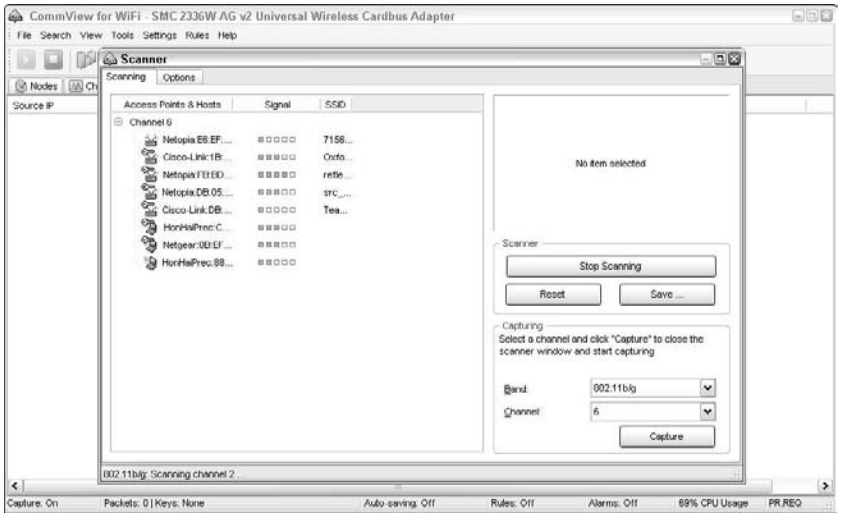


Figure 9-11: Using CommView for WiFi to spot wireless systems that don't belong.

My test network for this example is small compared to what you might see, but you get the idea of how an odd system can stand out.

WLANs set up in ad-hoc (or peer-to-peer) mode allow wireless clients to communicate directly with one another without having to pass through an AP. These types of WLANs operate outside the normal wireless security controls and can cause serious security issues beyond the normal 802.11 vulnerabilities. A good way to detect these rogue networks is to use NetStumbler.

You can use just about any WLAN analyzer to find unauthorized ad-hoc devices on your network. If you come across quite a few ad-hoc systems, such as those devices listed in AirMagnet WiFi Analyzer's STA section on its main screen, as shown in Figure 9-12, this could be a good indication that one (or several) people are running unprotected wireless systems or at least have ad-hoc wireless enabled. Either way, they're potentially putting your network and information at risk.

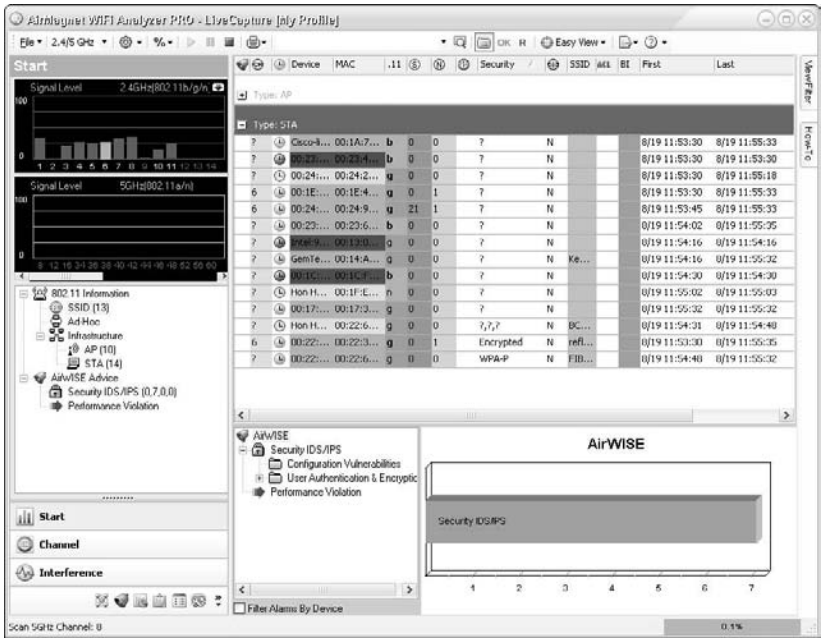


Figure 9-12: AirMagnet showing several unauthorized ad-hoc clients.

You can also use the handheld Digital Hotspotter I mention earlier in this chapter or even a wireless intrusion prevention system (IPS) to search for beacon packets in which the ESS field is not equal to 1.

Walk around your building or campus (*warwalk*, if you will) to perform this test to see what you can find. Physically look for devices that don't belong and keep in mind that a well-placed AP or WLAN client that's turned off won't show up in your network analysis tools. Search near the outskirts of the building or near any publicly accessible areas. Scope out boardrooms and the offices of upper-level managers for any unauthorized devices. These places are typically off-limits but often are used as locations for hackers to set up rogue APs.

When searching for unauthorized wireless devices on your network, keep in mind that you might be picking up signals from nearby offices or homes. Therefore, if you find something, don't immediately assume it's a rogue

device. One way to figure out whether a device is in a nearby office or home is by the strength of the signal you detect — devices outside your office *should* have a weaker signal than those inside. AirMagnet WiFi Analyzer has a neat way to monitor the signal strength of wireless devices you stumble onto. Figure 9-13 is a screenshot of AirMagnet's "Geiger counter" interface showing the relative signal strength of APs you find when warwalking.

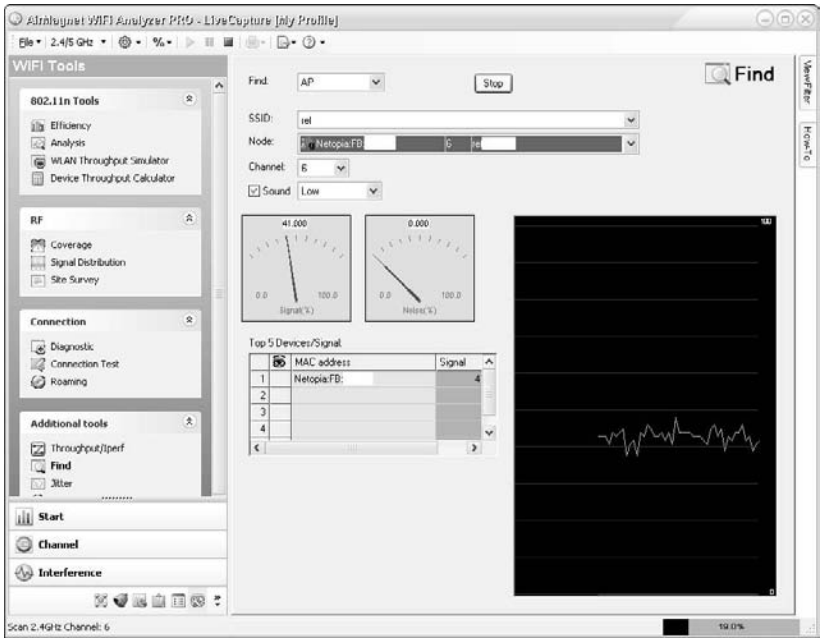


Figure 9-13: Using AirMagnet's WiFi Analyzer to monitor the signal strength of nearby wireless systems.

Using a WLAN analyzer in this way helps narrow the location and prevent false alarms in case you detect legitimate neighboring wireless devices.

A good way to determine whether an AP you discover is attached to your wired network is to perform reverse ARPs to map IP addresses to MAC addresses. You can do this at a command prompt by using the `arp -a` command and simply comparing IP addresses with the corresponding MAC address to see if you have a match.

Also, keep in mind that WLANs authenticate the wireless devices, not the users. Hackers can use this to their advantage by gaining access to a wireless client via remote-access software, such as telnet or SSH, or by exploiting a known application or OS vulnerability. After they do that, they potentially have full access to your network.

Countermeasures against rogue wireless devices

The only way to detect rogue APs and wireless hosts on your network is to monitor your WLAN proactively (say monthly, weekly, or using a wireless IPS, in real time), looking for indicators that wireless clients or rogue APs might exist. But if rogue APs or clients don't show up in NetStumbler or in your client manager software, that doesn't mean you're off the hook. You might also need to break out the WLAN analyzer, wireless IPS, or other network management application.

Depending on your AP, a couple of configuration changes might keep hackers from carrying out these hacks against you:

- If possible, increase your wireless beacon broadcast interval to the maximum setting, which is around 65,535 milliseconds (roughly 66 seconds). This can help hide the AP from hackers who are wardriving or walking by your building quickly. Be sure to test this first, though, because it might create other unintended consequences, such as legitimate wireless clients not being able to connect to your network.
- Disable probe responses to prevent your AP from responding to such requests.

Use personal firewall software, such as the free Windows Firewall that's built into Windows, on all wireless hosts to prevent unauthorized remote access to your network.

Finally, don't forget the holy grail of information security: user education. Ensuring that security is always on the top of everyone's mind can go further than just about any other measure when it comes to safe usage of wireless networks.

MAC spoofing

A very common defense for wireless networks is Media Access Control (MAC) address controls. This is where you configure your APs to allow only wireless clients with known MAC addresses to connect to the network. Consequently, a very common hack against wireless networks is MAC address spoofing.

The bad guys can easily spoof MAC addresses in UNIX, by using the `ifconfig` command, and in Windows, by using the SMAC utility, as I describe in Chapter 8. However, like WEP and WPA, MAC-address-based

access controls are another layer of protection and better than nothing at all. If someone spoofs one of your MAC addresses, the only way to detect malicious behavior is to spot the same MAC address being used in two or more places on the WLAN, which can be tricky.

One simple way to determine whether an AP is using MAC address controls is to try to associate with it and obtain an IP address via DHCP. If you can get an IP address, then the AP doesn't have MAC address controls enabled.

The following steps outline how you can test your MAC address controls and demonstrate just how easy they are to circumvent:

1. **Find an AP to attach to.**

   This can be done by simply loading NetStumbler, as shown in Figure 9-14.

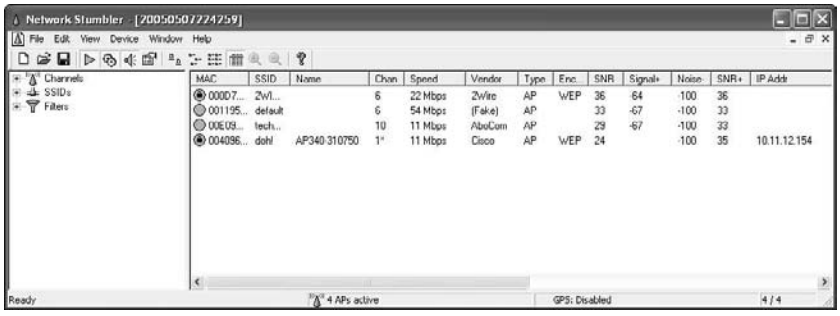


Figure 9-14: Finding an accessible AP via NetStumbler.

   In this test network, the AP with the SSID of *doh!* is the one I want to test. Note the MAC address of this AP as well. This will help you make sure you're looking at the right packets in the steps that follow. Although I've "hidden" most of the MAC address of this AP for the sake of privacy, let's just say its MAC address is 00:40:96:FF:FF:FF. Also, notice in Figure 9-14 that NetStumbler was able to determine the IP address of the AP. Getting an IP address will help you confirm that you're on the right wireless network.

2. **Using a WLAN analyzer, look for a wireless client sending a probe request packet to the broadcast address or the AP replying with a probe response.**

   You can set up a filter in your analyzer to look for such frames, or simply capture packets and just browse through looking for the AP's MAC address, which you noted in Step 1. Figure 9-15 shows what the Probe Request and Probe Response packets look like.

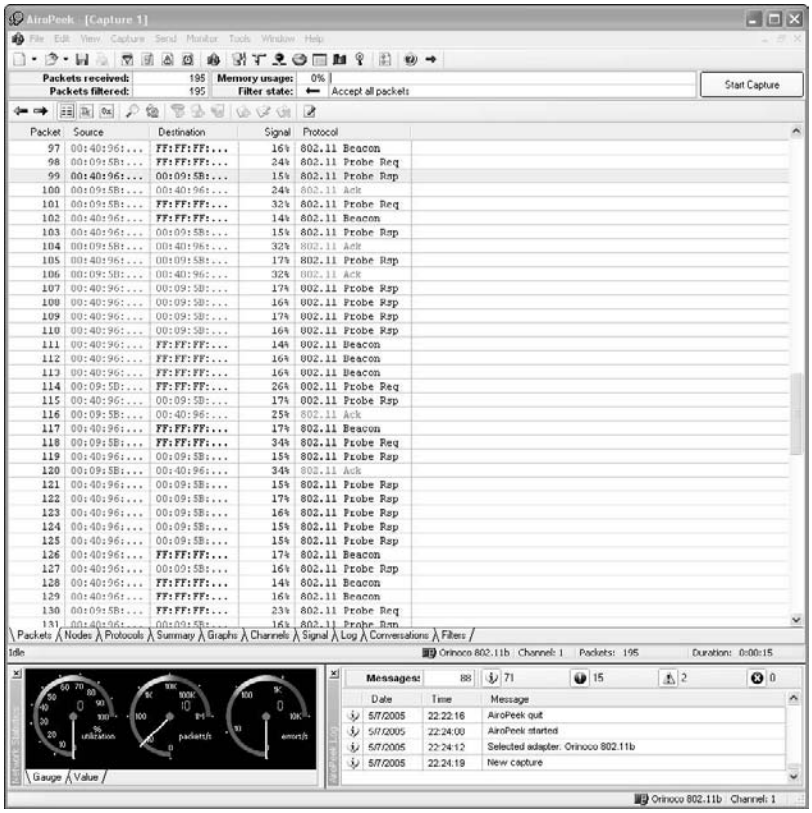


Figure 9-15: Looking for the MAC address of a wireless client on the network being tested.

Note that the wireless client (again for privacy, let's say its full MAC address is 00:09:5B:FF:FF:FF) first sends out a probe request to the broadcast address (FF:FF:FF:FF:FF:FF) in packet number 98. The AP with the MAC address I'm looking for replies with a Probe Response to 00:09:5B:FF:FF:FF, confirming that this is indeed a wireless client on the network for which I'll be testing MAC address controls.

3. **Change your test computer's MAC address to that of the wireless client's MAC address you found in Step 2.**

   In UNIX and Linux, you can change your MAC address very easily by using the `ifconfig` command as follows:

   a. *Log in as root and then disable the network interface.*

      Insert the network interface number that you want to disable (typically `wlan0` or `ath0`) into the command, like this:

```
[root@localhost root]# ifconfig wlan0 down
```

b. Enter the new MAC address you wish to use.

Insert the fake MAC address and the network interface number like this:

```
[root@localhost root]# ifconfig wlan0 hw ether
     01:23:45:67:89:ab
```

The following command also works in Linux:

```
[root@localhost root]# ip link set wlan0 address
     01:23:45:67:89:ab
```

c. Bring the interface back up with this command:

```
[root@localhost root]# ifconfig wlan0 up
```

If you change your Linux MAC addresses often, you can use a more feature-rich utility called MAC Changer (`www.alobbs.com/macchanger`).

In Windows, you might be able to change your MAC addresses in your wireless NIC properties via My Network Places. However, if you don't like editing the registry or prefer to have an automated tool, you can use a neat and inexpensive tool created by KLC Consulting called SMAC (available at `www.klcconsulting.net/smac`). To change your MAC address, you can use the steps I outline in Chapter 8.

When you're done, SMAC will show something similar to the screen capture in Figure 9-16.

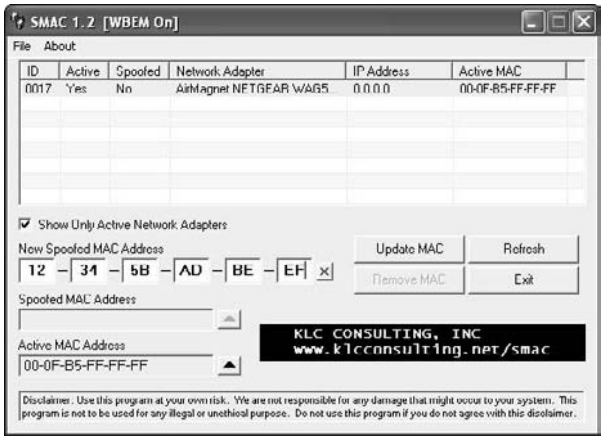


Figure 9-16: SMAC showing a spoofed MAC address.

To reverse any of the above MAC address changes, simply reverse the steps performed and then delete any data you created.

Note that APs, routers, switches, and the like might detect when more than one system is using the same MAC address on the network (that is, yours and the client that you're spoofing). You might have to wait until that system is no longer on the network; however, I rarely see any issues spoofing MAC addresses in this way, so you probably won't have to do anything.

4. **Ensure that your wireless NIC is configured for the appropriate SSID.**

   For this example, I used the Netgear Smart Wizard utility to set the SSID to *doh!*, as shown in Figure 9-17.

Figure 9-17: Ensure your SSID is correctly set.

Even if your network is running WEP or WPA, you can still test your MAC address controls. You just need to enter your encryption key(s) before you can connect.

5. **Obtain an IP address on the network.**

   You can do this by rebooting or disabling/enabling your wireless NIC. However, you can do it manually by running `ipconfig /renew` at a Windows command prompt or by manually entering a known IP address in your wireless network card's network properties.

6. **Confirm that you're on the network by pinging another host or browsing the Internet.**

   In this example, I could ping the AP (10.11.12.154) or simply load my favorite Web browser to see whether I can access the Internet.

That's all there is to it! You've circumvented your wireless network's MAC address controls in six simple steps. Piece of cake!

Countermeasures against MAC spoofing

The easiest way to prevent the circumvention of MAC address controls and subsequent unauthorized attachment to your wireless network is to enable WPA, or ideallyWPA2. Another way to control MAC spoofing is by using a wireless IPS. This second option is certainly more costly, but it could be well worth the money when you consider the other proactive monitoring and blocking benefits such a system would provide.

Queensland DoS attack

A relatively new and mostly unheard-of attack against the 802.11 protocol was discovered in May of 2004 by researchers at the Queensland University of Technology's Information Security Research Centre (`www.kb.cert.org/vuls/id/106678`). This "Queensland" attack, also referred to as the Clear Channel Assessment attack, affects the Direct Sequence Spread Spectrum function that works as part of 802.11's Carrier Sense Multiple Access/Collision Avoidance (CSMA/CA) protocol that manages the wireless communications medium.

Wireless systems (clients, APs, and so on) use CSMA/CA to determine whether the wireless medium is ready and the system can transmit data. The Queensland attack exploits the Clear Channel Assessment (CCA) function within CSMA/CA and makes it appear that the airwaves are busy, effectively preventing any other wireless system from transmitting. This is accomplished by placing a wireless NIC in continuous transmit mode.

With the right tool, this attack is relatively simple to execute. It can wreak havoc on a wireless network, effectively bringing it to its knees. There's very little that can be done about it, especially if the attacker's signal is more powerful than that of your wireless systems.

All it takes to run this attack is to find an old D-Link DWL-650 wireless NIC (via eBay or elsewhere) combined with the old Prism chipset testing program called Prism Test Utility (`PrismTestUtil322.exe`). This program was previously available for public download on Intersil's Web site and is still available if you dig around on the Internet (try Googling the filename above). This attack can just as easily be carried out with other custom software or hardware tweaking as well. No need for screen captures here. Simply stated, before you put a wireless NIC in continuous transmit mode, you have wireless signals. After you exploit this weakness — wireless all gone!

This test can be hazardous to your wireless network's health! Run this test only in a controlled environment to test a wireless IPS and in a way that doesn't affect other people's wireless networks located nearby.

Countermeasures against DoS attacks

The only *potential* countermeasure against this and other wireless DoS attacks is the installation and usage of a wireless IPS on your 802.11b/g network. Otherwise, wireless technologies that use frequency hopping spread spectrum (FHSS) or orthogonal frequency division multiplexing (OFDM) — such as 802.11a, 802.11n, and technically 802.11g running over 20Mbps — are the way to go.

Physical security problems

Various physical security vulnerabilities can result in physical theft, the reconfiguration of wireless devices, and the capturing of confidential information. You should look for the following security vulnerabilities when testing your systems:

- APs mounted on the outside of a building and accessible to the public.
- Poorly mounted antennas — or the wrong types of antennas — that broadcast too strong a signal and that are accessible to the public. You can view the signal strength in NetStumbler, your wireless client manager, or one of the commercial tools I mention earlier in this chapter.

These issues are often overlooked because of rushed installations, improper planning, and lack of technical knowledge, but they can come back to haunt you.

Countermeasures against physical security problems

Ensure that APs, antennas, and other wireless and network infrastructure equipment are locked away in secure closets, ceilings, or other places that are difficult for a would-be intruder to access physically. Terminate your APs outside any firewall or other network perimeter security devices — or at least in a DMZ — whenever possible. If you place the wireless equipment inside your secure network, it can negate any benefits you would get from your perimeter security devices, such as your firewall.

If wireless signals are propagating outside your building where they don't belong, either

- Turn down the transmit power setting of your AP.
- Use a smaller or different antenna (semidirectional or directional) to decrease the signal.

Some basic planning helps prevent these vulnerabilities.

Vulnerable wireless workstations

Wireless workstations have tons of security vulnerabilities — from weak passwords to unpatched security holes to the storage of WEP and WPA encryption keys locally. Most of the well-known wireless client vulnerabilities have been patched by their respective vendors, but you never know if all your wireless systems are running the latest (and usually safest) versions of operating systems, wireless client software, and other software applications.

In addition to using the wireless client, stumbling, and network analysis software I mention earlier in this chapter, you should also search for wireless client vulnerabilities by using various vulnerability testing tools, such as GFI LANguard, QualysGuard, and Acunetix Web Vulnerability Scanner.

These programs aren't wireless-specific, but they might turn up vulnerabilities in your wireless computers that you might not have discovered or thought about testing otherwise. I cover operating system and application vulnerabilities as well as using the tools in the preceding list in Parts IV and V of this book.

Countermeasures against vulnerable wireless workstations

You can implement the following countermeasures to keep your workstations from being used as entry points into your WLAN:

- **Regularly perform vulnerability assessments on your wireless workstations, in addition to other network hosts.**
- **Apply the latest vendor security patches and enforce strong user passwords.**

- **Use personal firewalls and endpoint security software on *all* wireless systems where possible,** including PDAs and smartphones to keep malicious intruders off those systems and out of your network.
- **Install anti-malware software.**

Default configuration settings

Similar to wireless workstations, wireless APs have many known vulnerabilities. The most common ones are default SSIDs and admin passwords. The more specific ones occur only on certain hardware and software versions that are posted in vulnerability databases and vendor Web sites. Many wireless systems *still* have WEP and WPA disabled by default as well.

Countermeasures against default configuration settings exploits

You can implement some of the simplest and most effective security countermeasures for WLANs — and they're all free:

- **Make sure that you change default admin passwords and SSIDs.**
- **At a minimum, enable WPA.** Ideally, you should use WPA2 with very strong pre-shared keys (PSKs) consisting of at least 20 random characters or use WPA/WPA2 in "enterprise" mode with a RADIUS server for host authentication.
- **Disable SSID broadcasting if you don't need this feature.**
- **Apply the latest firmware patches for your APs and WLAN cards.** This countermeasure helps to prevent various vulnerabilities to prevent the exploitation of publicly known holes related to management interfaces on APs and client management software on the clients.

Part IV
Hacking Operating Systems

In this part . . .

Now that you're past the network level, it's time to get down to the nitty-gritty — those fun operating systems you use on a daily basis and have come to both love and hate. I definitely don't have enough room in this book to cover every operating system version or even every operating system vulnerability, but I certainly hit the important parts — especially the ones that aren't easily fixed with patches.

This part starts by looking at the most widely used (and picked on) operating system — Microsoft Windows. From Windows NT to Windows 7, I show you some of the best ways to attack these operating systems and secure them from the bad guys. This part then looks at Linux and its less publicized (yet still major) security flaws. Many of the hacks and countermeasures I cover can apply to many other flavors of UNIX as well. This part then moves on to the tried-and-true Novell NetWare operating system — perhaps the most secure OS in this lineup, though it's still not vulnerability-free, as many Novell die-hards like to believe. I cover the major issues along with solid countermeasures you can implement to keep your mighty NetWare boxes secure and still (mostly) reboot-free.

Chapter 10

Windows

In This Chapter

- Port scanning Windows systems
- Gleaning Windows information without logging in
- Windows 7 security pros and cons
- Exploiting Windows vulnerabilities
- Minimizing Windows security risks

Microsoft Windows (with such versions as Windows XP, Windows Server 2003, Windows Vista, and Windows 7) is the most widely used operating system (OS) in the world. It's also the most widely abused. Is this because Microsoft doesn't care as much about security as other OS vendors? The short answer is no. Sure, numerous security flaws were overlooked — especially in the Windows NT days — but Microsoft products are so pervasive throughout today's networks, Microsoft is the easiest vendor to pick on; therefore Microsoft products often end up in the bad guys' crosshairs. The one positive about hackers is that they're driving the requirement for better security!

Many of the security flaws in the headlines aren't new. They're variants of vulnerabilities that have been around for a long time in UNIX and Linux, such as the remote procedure call (RPC) vulnerabilities that the Blaster worm exploited. You've heard the saying, "The more things change, the more they stay the same." That applies here, too. Most Windows attacks are preventable if the patches are properly applied. Thus, poor security management is often the real reason Windows attacks are successful, yet Microsoft takes the blame and must carry the burden.

In addition to the password attacks I cover in Chapter 7, many other attacks are possible against a Windows-based system. Tons of information can be extracted from Windows by simply connecting to the system across a network and using tools to pull out the information. Many of these tests don't even require you to be authenticated to the remote system. All someone with

malicious intent needs to find on your network is a vulnerable Windows computer with a default configuration that's not protected by such measures as a personal firewall and the latest security patches.

When you start poking around on your network, you might be surprised at how many of your Windows-based computers have security vulnerabilities. Furthermore, you'll be even more surprised at just how easy it is to exploit vulnerabilities to gain complete remote control of Windows by using a tool such as Metasploit. After you connect to a Windows system and have a valid username and password (by knowing it or deriving it by using the password-cracking techniques in Chapter 7 or other techniques outlined in this chapter), you can dig deeper and exploit other aspects of Windows.

This chapter shows you how to test for some of the most critical attacks against the Windows OS and outlines countermeasures to make sure your systems are secure.

Windows Vulnerabilities

Given the ease of use of Windows, its enterprise-ready Active Directory service, and the feature-rich .NET development platform, many organizations have moved to the Microsoft platform for their networking and computing needs. Many businesses — especially the small- to medium-sized ones — depend solely on the Windows OS for network usage. Many large organizations run critical servers, such as Web servers and database servers, on the Windows platform as well. If security vulnerabilities aren't addressed and managed properly, they can bring a network or an entire organization to its knees.

When Windows and other Microsoft software are attacked — especially by a widespread Internet-based worm or virus — hundreds of thousands of organizations and millions of computers are affected. Many well-known attacks against Windows can lead to

- Leakage of sensitive information, including files containing healthcare information and credit card numbers
- Passwords being cracked and used to carry out other attacks
- Systems taken completely offline by denial of service (DoS) attacks
- Full remote control being obtained
- Entire databases being corrupted or deleted

When unsecure Windows-based systems are attacked, serious things can happen to a tremendous number of computers around the world.

Choosing Tools

Literally hundreds of Windows hacking and testing tools are available. The key is to find a set of tools that can do what you need and that you're comfortable using.

Many security tools — including some of the tools in this chapter — work with only certain versions of Windows. The most recent version of each tool in this chapter is compatible with Windows NT, Windows 2000, Windows XP, and Windows Server 2003. I've even found many tools to be compatible with Windows 7, which happens to be the OS I use.

The more security tools and other power-user applications you install in Windows — especially programs that tie into the network drivers and TCP/IP stack — the more unstable Windows becomes. I'm talking about slow performance, blue screens of death, and general instability issues. Unfortunately, often the only fix is to reinstall Windows and all your applications. After rebuilding my laptop every few months, I finally wised up and bought a copy of VMware and a dedicated computer that I can junk up with testing tools without worrying about it affecting my ability to get my other work done. (Ah, the memories of those DOS and Windows 3.*x* days when things were much simpler!)

Free Microsoft tools

You can use the following free Microsoft tools to test your systems for various security weaknesses.

- **Built-in Windows programs** (Windows 9*x* and later versions) for NetBIOS and TCP/UDP service enumeration, such as:
  - nbtstat for gather NetBIOS name table information
  - netstat for displaying open ports on the local Windows system
  - net for running various network-based commands, including viewing shares on remote Windows systems and adding user accounts after you gain a remote command prompt via Metasploit
- **Microsoft Baseline Security Analyzer** (`www.microsoft.com/technet/security/tools/mbsahome.mspx`) to test for missing patches and basic Windows security settings
- **Sysinternals** (`http://technet.microsoft.com/en-us/sysinternals/default.aspx`) to poke, prod, and monitor Windows services, processes, and resources both locally and over the network

All-in-one assessment tools

All-in-one tools perform a wide variety of security tests, including

- Port scanning
- OS fingerprinting
- Basic password cracking
- Detailed vulnerability mappings of the various security weaknesses that the tools find on your Windows systems

I use the following tools in my work with very good results:

- **GFI LANguard** (`www.gfi.com/lannetscan`)
- **QualysGuard** (`www.qualys.com`)

Qualys's managed service/application service provider/software as a service (whatever term you want to use these days) is very easy to use (simply log in to the interface, give it the IP addresses to scan, and tell it to go) and has very detailed and accurate vulnerability testing — it's my all-time favorite for network/OS vulnerability testing.

Task-specific tools

The following tools perform one or two specific tasks. These tools provide detailed security assessments of your Windows systems and insight that you might not otherwise get from all-in-one assessment tools:

- **Metasploit** (`www.metasploit.com`) for exploiting vulnerabilities that such tools as QualysGuard and Nessus (`www.nessus.org`) discover to obtain remote command prompts, add users, and much more.
- **ShareEnum** (`http://technet.microsoft.com/en-us/sysinternals/bb897442.aspx`) for share enumeration.
- **SuperScan** (`www.foundstone.com/us/resources/proddesc/superscan.htm`) for TCP port scanning, ping sweeps, and share enumeration.
- **TCPView** (`http://technet.microsoft.com/en-us/sysinternals/bb897437.aspx`) to view TCP and UDP session information.
- **Winfo** (`www.ntsecurity.nu/toolbox/winfo`) for null session enumeration to gather such configuration information as security policies, local user accounts, and shares.

Windows XP SP2 and later versions, as well as Windows Server 2003 SP1 and later versions, have a new "undocumented feature" that can (and will) severely limit your network scanning speeds: Only ten half-open TCP connections can be made at a time. If you think your system might be affected by this, check out the Event ID 4226 Patcher tool (`www.lvllord.de`) for a hack to run on the Windows TCP/IP stack that will allow you to adjust the TCP half-open connections setting to a more realistic number. The default is to change it to 50, which seems to work well.

Be forewarned that Microsoft doesn't support this hack. Having said that, I haven't had any trouble with this hack at all. Disabling the Windows Firewall (or other third-party firewall) can help speed things up, too.

Information Gathering

When you assess Windows vulnerabilities, start by scanning your computers to see what the bad guys can see.

The exploits in this chapter were run against Windows from inside a firewall. Unless I point out otherwise, all the tests in this chapter can be run against all versions of the Windows OS. The attacks in this chapter are significant enough to warrant testing for, regardless of your current setup. Your results might vary from mine depending on the specific version of Windows, patch levels, and other system hardening you've done.

System scanning

A few straightforward processes can identify weaknesses in Windows systems.

Testing

Start gathering information about your Windows systems by running an initial port scan:

1. **Run basic scans to find which ports are open on each Windows system:**
    - Scan for TCP ports with a port scanning tool, such as SuperScan. The SuperScan results in Figure 10-1 show several potentially vulnerable ports open on a Windows Server 2003 system, including those for a Web server (port 80), and the ever-popular — and easily hacked — NetBIOS (port 139).

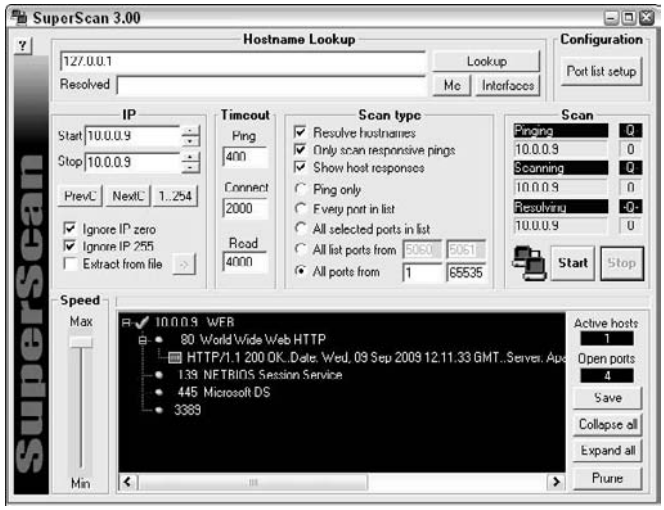


Figure 10-1: Port scanning a Windows Server 2003 system with SuperScan.

2. **Perform OS enumeration (such as scanning for shares and specific OS versions) by using an all-in-one assessment tool, such as LANguard.**

   Figure 10-2 shows a LANguard scan that reveals the server version, vulnerabilities, open ports, and more.

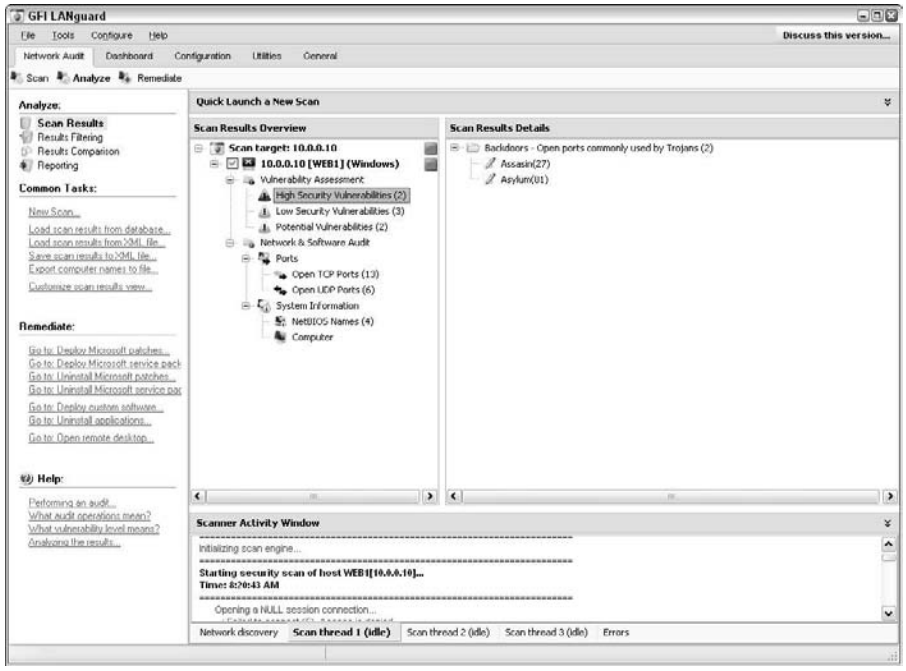


Figure 10-2: Gathering detailed vulnerabilities of a Windows 2000 Server with LANguard.

If you need to quickly identify the specific version of Windows that's running, you can use Nmap (`http://nmap.org/download.html`) with the `-O` option, as shown in Figure 10-3.

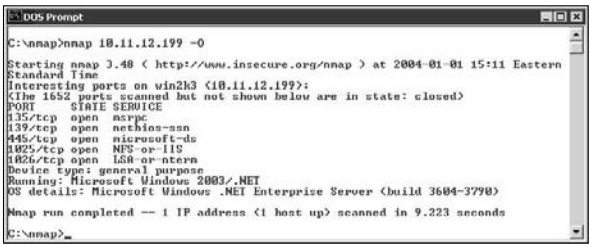


Figure 10-3: Using Nmap to determine the Windows version.

Other OS fingerprinting tools are available, but I've found Nmap to be the most accurate.

3. **Determine potential security vulnerabilities.**

   This is subjective and might vary from system to system, but what you want to look for are interesting services and applications and proceed from there.

Countermeasures against system scanning

You can prevent an external attacker or malicious internal user from gathering certain information about your Windows systems by implementing the proper security settings on your network and on the Windows hosts. You have the following options:

- Use a network firewall.
- Use the Windows Firewall or other personal firewall software on each system. You want to block the Windows networking ports for RPC (port 135) and NetBIOS (ports 137–139 and 445).
- Disable unnecessary services so that they don't appear when a connection is made.

NetBIOS

You can gather Windows information by poking around with NetBIOS (Network Basic Input/Output System) functions and programs. NetBIOS allows applications to make networking calls and communicate with other hosts within a LAN.

These Windows NetBIOS ports can be compromised if they aren't properly secured:

- **UDP ports for network browsing:**
  - Port 137 (NetBIOS name services)
  - Port 138 (NetBIOS datagram services)
- **TCP ports for Server Message Block (SMB):**
  - Port 139 (NetBIOS session services)
  - Port 445 (runs SMB over TCP/IP without NetBIOS)

Hacks

The hacks described in the following two sections can be carried out on unprotected systems running NetBIOS.

Unauthenticated enumeration

When you're performing your unauthenticated enumeration tests, you can gather configuration information about the local or remote systems two ways:

- All-in-one scanners, such as LANguard or QualysGuard
- The nbtstat program that's built in to Windows (nbtstat stands for NetBIOS over TCP/IP Statistics)

Figure 10-4 shows information that you can gather from a Windows 7 system with a simple nbtstat query.

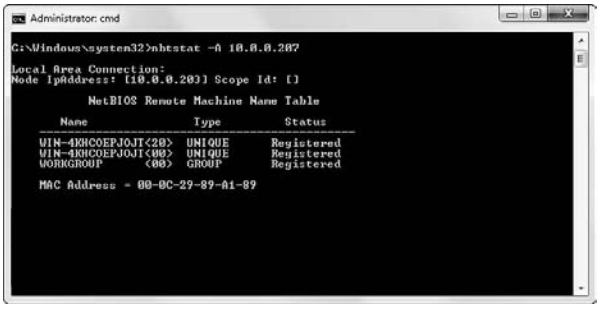


Figure 10-4: Using nbtstat to gather information on a Windows 7 system.

nbtstat shows the remote computer's NetBIOS name table, which you gather by using the `nbtstat -A` command. This displays the following information:

- Computer name
- Domain name
- Computer's MAC address

When running nbtstat against a Windows NT or Windows 2000 server, you might even glean the ID of the user who's currently logged in.

An advanced program such as LANguard isn't necessary to gather this basic information from a Windows system. However, the graphical interface offered by commercial software such as this presents its findings in a prettier fashion and is often much easier to use. Additionally, you have the benefit of gathering the information you need with one tool.

Shares

Windows uses network shares to *share* certain folders or drives on the system so other users can access them across the network. Shares are easy to set up and work very well. However, they're often misconfigured, allowing hackers and other unauthorized users to access information they shouldn't be able to get to. You can search for Windows network shares by using the Share Finder tool built in to LANguard. This tool scans an entire range of IP addresses, looking for Windows shares, as shown in Figure 10-5.

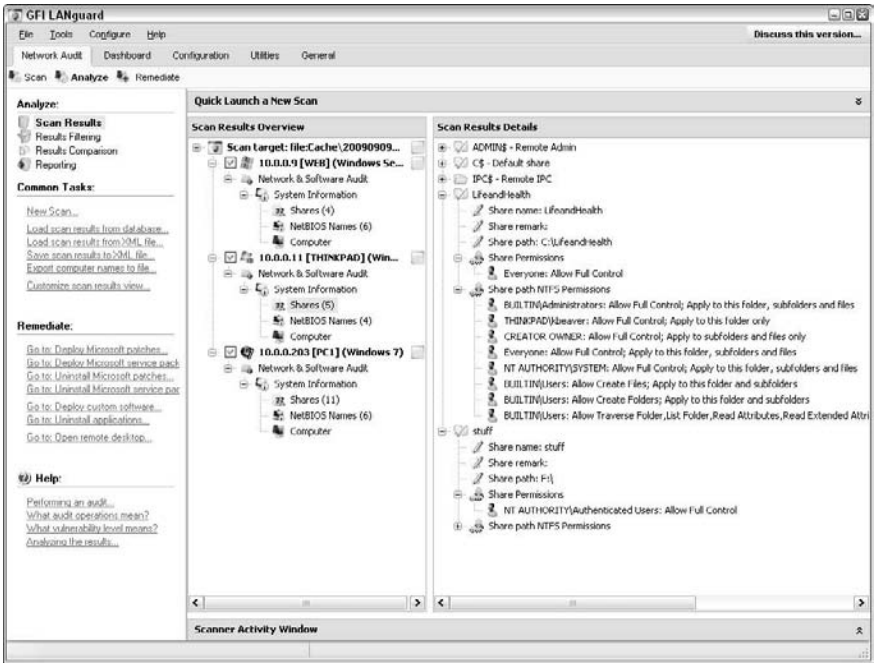


Figure 10-5: Using LANguard to scan your network for Windows shares.

The shares displayed in Figure 10-5 are just what malicious insiders are looking for because the share names give a hint of what type of files might be accessible if they connect to the shares. After the bad guys discover these shares, they're likely to dig a little further to see whether they can browse the files within the shares. I cover shares and rooting out sensitive information on network shares and other storage devices in Chapter 15.

Countermeasures against NetBIOS attacks

You can implement the following security countermeasures to minimize NetBIOS and NetBIOS over TCP/IP attacks on your Windows systems:

- Use a network firewall.
- Use the Windows Firewall or other personal firewall software on each system.
- Disable NetBIOS — or at least Windows File and Printer Sharing.

  Disabling NetBIOS might not be practical in a network where users and applications depend on file sharing or in a mixed environment where older Windows 2000 and NT systems rely on NetBIOS for file and printer sharing.
- Educate your users on the dangers of enabling file shares for everyone to access. I cover these risks in detail in Chapter 15.

Hidden shares — those with a dollar sign ($) appended to the end of the share name — don't really help hide the share name. Any of the tools I've mentioned can see right through this form of security by obscurity.

Null Sessions

A well-known vulnerability within Windows can map an anonymous connection (or *null session*) to a hidden share called IPC$ (which stands for interprocess communication). This attack method can be used to:

- Gather Windows host configuration information, such as user IDs and share names
- Edit parts of the remote computer's registry

Although Windows Server 2003/2008, Windows XP, Windows Vista, and Windows 7 don't allow null session connections by default, Windows 2000 Server and NT Server do — and plenty of those systems are still around to cause problems on most networks.

Although later versions of Windows are much more secure than their predecessors, don't assume that all's well in Windows land. I can't tell you how many times I see supposedly secure Windows installations "tweaked" to accommodate an application or other business need that happens to facilitate exploitation.

Mapping

To map a null session, follow these steps for each Windows computer to which you want to map a null session:

1. **Format the basic `net` command, like this:**

   ```
   net use \\host_name_or_IP_address\ipc$ "" "/user:"
   ```

   The `net` command to map null sessions requires these parameters:

   - `net` (the built-in Windows *network* command) followed by the `use` command
   - IP address or hostname of the system to which you want to map a null connection
   - A blank password and username

     The blanks are why it's called a *null* connection.

2. **Press Enter to make the connection.**

   Figure 10-6 shows an example of the complete command when mapping a null session. After you map the null session, you should see the message `The command completed successfully`.

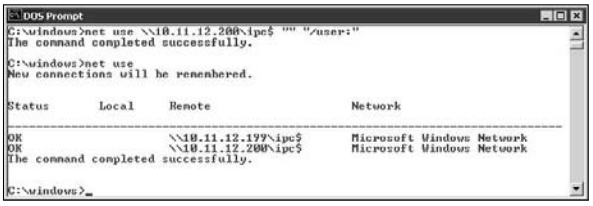


Figure 10-6: Mapping a null session to a vulnerable Windows system.

To confirm that the sessions are mapped, enter this command at the command prompt:

```
net use
```

As shown in Figure 10-6, you should see the mappings to the IPC$ share on each computer to which you're connected.

Gleaning information

With a null session connection, you can use other utilities to gather critical Windows information remotely. Dozens of tools can gather this type of information.

You — like a hacker — can take the output of these enumeration programs and attempt (as an unauthorized user) to:

- Crack the passwords of the users found. (See Chapter 7 for more on password cracking.)
- Map drives to the network shares.

You can use the following applications for system enumeration against server versions of Windows prior to Server 2003 as well as Windows XP.

net view

The `net view` command (see Figure 10-7) shows shares that the Windows host has available. You can use the output of this program to see information that the server is advertising to the world and what can be done with it, including:

- Share information that a hacker can use to attack your systems, such as mapping drives and cracking share passwords.
- Share permissions that might need to be removed, such as the permission for the Everyone group, to at least see the share on Windows NT and Windows 2000 systems.

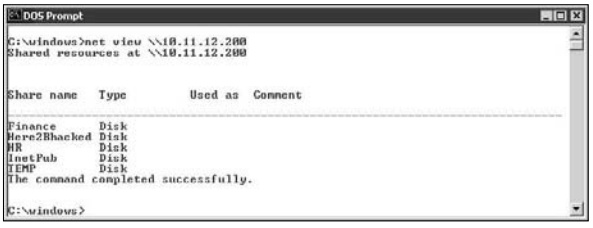


Figure 10-7: net view displays drive shares on a remote Windows host.

Configuration and user information

Winfo and DumpSec can gather useful information about users and configurations, such as

- Windows domain to which the system belongs
- Security policy settings
- Local usernames
- Drive shares

Your preference might depend on whether you like graphical interfaces or a command line:

- Winfo (`www.ntsecurity.nu/toolbox/winfo`) is a command-line tool.

Because Winfo is a command-line tool, you can create batch (script) files that automate the enumeration process. The following is an abbreviated version of Winfo's output of a Windows NT server, but you can collect the same information from other Windows systems:

```
Winfo 2.0 - copyright (c) 1999-2003, Arne Vidstrom
          - http://www.ntsecurity.nu/toolbox/winfo/
SYSTEM INFORMATION:
 - OS version: 4.0
PASSWORD POLICY:
 - Time between end of logon time and forced logoff: No forced logoff
 - Maximum password age: 42 days
 - Minimum password age: 0 days
 - Password history length: 0 passwords
 - Minimum password length: 0 characters
USER ACCOUNTS:
 * Administrator
   (This account is the built-in administrator account)
 * doctorx
 * Guest
   (This account is the built-in guest account)
 * IUSR_WINNT
 * kbeaver
 * nikki
SHARES:
 * ADMIN$
    - Type: Special share reserved for IPC or administrative share
 * IPC$
    - Type: Unknown
 * Here2Bhacked
    - Type: Disk drive
 * C$
    - Type: Special share reserved for IPC or administrative share
 * Finance
    - Type: Disk drive
 * HR
    - Type: Disk drive
```

This information cannot be gleaned from a default installation of Windows Server 2003, Windows XP, Windows Vista, or Windows 7.

You can peruse the output of such tools for user IDs that don't belong on your system, such as

- Ex-employee accounts that haven't been disabled
- Potential backdoor accounts that a hacker might have created

If attackers get this information, they can attempt to exploit potentially weak passwords and log in as those users.

NetUsers

The NetUsers tool (`www.systemtools.com/free.htm`) can show who has logged into a remote Windows computer. You can see such information as

- Abused account privileges
- Users currently logged into the system

Figure 10-8 shows the history of local logins of a remote Windows workstation.

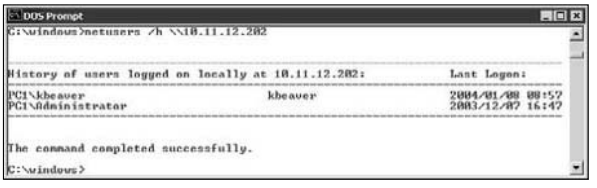


Figure 10-8: The NetUsers tool.

This information can help you track, for auditing purposes, who's logging into a system. Unfortunately, this information can be useful for hackers when they're trying to figure out what user IDs are available to crack. They might even determine the system's daily use if the users IDs are descriptive, such as *backup* (for a backup server) or *devuser* (for a development user).

Countermeasures against null session hacks

If it makes good business sense and the timing is right, upgrade to the more secure Windows Server 2003, Windows Server 2008, and Windows 7. They don't have the vulnerabilities described in the following list.

You can easily prevent null session connection hacks by implementing one or more of the following security measures:

- Block NetBIOS on your Windows server by preventing these TCP ports from passing through your network firewall or personal firewall:
  - 139 (NetBIOS sessions services)
  - 445 (runs SMB over TCP/IP without NetBIOS)
- Disable File and Printer Sharing for Microsoft Networks in the Properties tab of the machine's network connection for those systems that don't need it.
- Restrict anonymous connections to the system. For Windows NT and Windows 2000 systems, you can set `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\LSA\RestrictAnonymous` to a DWORD value as follows:
  - *None:* This is the default setting.
  - *Rely on Default Permissions (Setting 0):* This setting allows the default null session connections.
  - *Do Not Allow Enumeration of SAM Accounts and Shares (Setting 1):* This is the medium security level setting. This setting still allows null sessions to be mapped to IPC$, enabling such tools as Walksam to garner information from the system.
  - *No Access without Explicit Anonymous Permissions (Setting 2):* This high security setting prevents null session connections and system enumeration.

  High security creates problems for domain controller communication and network browsing, so be careful!

Microsoft Knowledge Base Article 246261 covers the caveats of using the high security setting for `RestrictAnonymous`. It's available on the Web at `http://support.microsoft.com/default.aspx?scid=KB;en-us;246261`.

For later versions of Windows, such as Windows Server 2003 and Windows 7, ensure that the Network Access "anonymous" components of the local or group security policy are set as shown in Figure 10-9.

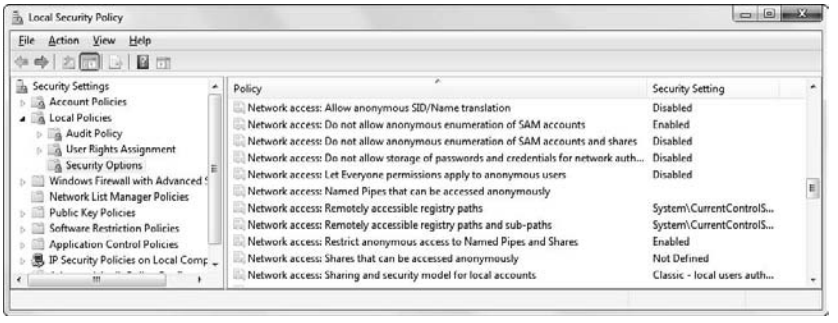


Figure 10-9: Default local security policy settings in Windows 7 that restrict null session connections.

Share Permissions

Windows *shares* — the available network drives that show up when browsing the network in My Network Places — are often misconfigured, allowing more people to have access to them than they should. The casual browser can exploit this security vulnerability, but a malicious insider gaining unauthorized access to a Windows system can result in serious security and compliance consequences, including the leakage of sensitive information and even the corruption or deletion of critical files.

Windows defaults

The default share permission depends on the Windows system version.

Windows 2000/NT

When creating shares in Windows NT and Windows 2000, the group Everyone is given Full Control access in the share by default for all files to

- Browse files
- Read files
- Write files

Anyone who maps to the IPC$ connection with a null session (as described in the previous section, "Null Sessions") is automatically made part of the Everyone group. This means that remote hackers can automatically gain Browse, Read, and Write access to a Windows NT or Windows 2000 server after establishing a null session.

Windows XP

In Windows XP and newer (Windows 2003 Server, Windows Vista, Windows 7), the Everyone group is given only Read access to shares. This is definitely an improvement over the defaults in Windows 2000 and Windows NT. However, you still might have situations in which you don't want the Everyone group to have Read access to a share.

Share permissions are different from file permissions. When creating shares, you have to set both. In current versions of Windows, this helps create hoops for casual users to jump through and discourage share creation but it's not foolproof. Unless you have your Windows desktops completely locked down, users can still share at will.

Testing

Assessing your share permissions is a good way to get an overall view of who can access what. This testing shows how vulnerable your network shares — and sensitive information — can be. You can find shares with default permissions and unnecessary access rights enabled. Trust me, they're everywhere!

The best way to test for share weaknesses is to log in to the Windows system via a standard local or domain user with no special privileges and run an enumeration program so you can see who has access to what.

LANguard has a built-in share finder tool for uncovering unprotected shares, as shown in Figure 10-10.

The Everyone group has full share and file access to the LifeandHealth share on the THINKPAD host. I see situations like this all the time where someone shares their local drive so others can access it. The problem is they often forget to remove the permissions and leave a gaping hole for a security breach. I outline how to uncover sensitive information in unstructured files on shares and other storage systems in Chapter 15.

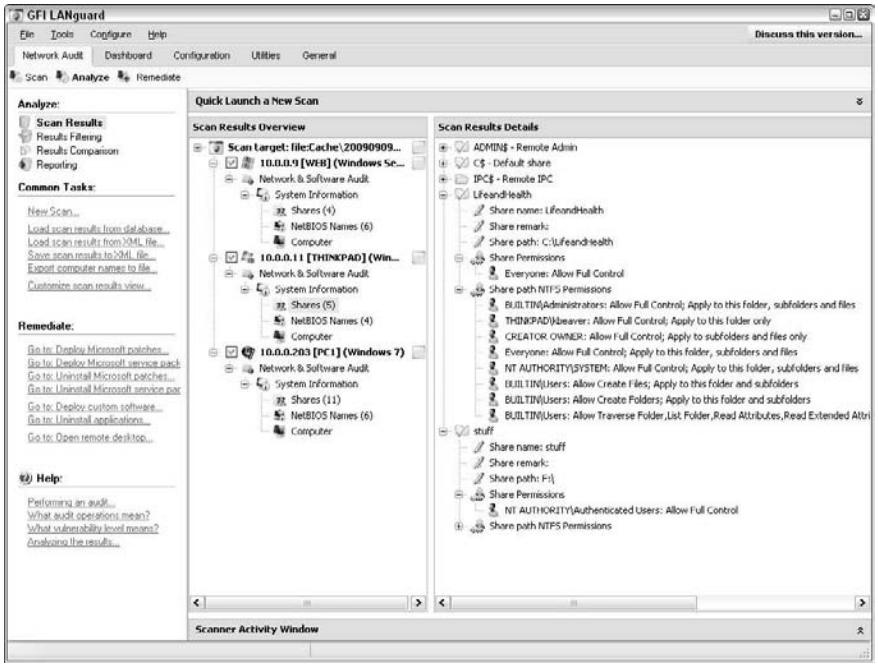


Figure 10-10: Using LANguard's share finder to seek out Windows shares.

Missing Patch Exploitation

It's one thing to poke and prod Windows to find vulnerabilities that might eventually lead to some good information — maybe system access. However, it's quite another to stumble across a vulnerability that will provide you with full and complete system access — all within 10 minutes or less. Well, it's no longer an empty threat that "arbitrary code" can be run on a system that *may* lead to a vulnerability exploitation. Now, with such tools as Metasploit, all it takes is one missing patch on one system to gain access and demonstrate how the entire network can be compromised. This is the ethical hacker's pot of gold.

Windows 7 security

With all these vulnerabilities, you might want to jump ship and move to Linux. But not so fast. Microsoft has made great strides with security in Windows 7. Although Windows Vista, like Windows Me, took a bunch of abuse and left an ugly scar on Microsoft, Vista did lay the groundwork for what's now the much improved Windows 7. I have to admit that when it comes to security, Microsoft has finally seen the light with this operating system. Windows 7 features include

- Windows Defender spyware protection that's enabled by default
- A beefed-up version of Windows Firewall that has both inbound and outbound protection to keep malware from doing bad things
- No local admin rights for regular users via User Account Control (UAC) that will keep users and malware from performing administrator-level functions to muck up the system
- Restricted services running with minimal privileges to minimize damage if they're compromised
- Network Access Protection (NAP) when used in conjunction with Windows Server 2008 that allows only "clean" systems to connect to the network
- Drive encryption via BitLocker
- Lots of privacy enhancements and security updates in Internet Explorer 8

Having run various scans and attacks against Windows 7 systems, I can say it's the most secure default installation of Windows I've seen. So, does all this mean that Windows 7 is immune to attack and abuse? Of course not. As long as the human element is involved in the software development, network administration, and end-user functions, people will continue to make mistakes that leave "windows" open for the bad guys to sneak through and carry out their attacks. In fact, numerous security updates for Windows 7 have already been released. Interestingly, the very day I'm writing this information a new attack against the SMB protocol in Windows Vista and potentially Windows 7 was announced that allows an attacker to execute code remotely on the system. This stuff is never ending! Furthermore, if your mobile Windows 7 systems are ever lost or stolen, they're just as vulnerable to the password attacks that I cover in Chapter 7 as any other version of Windows. The key is to make sure you never let your guard down.

Even with all the strict policies and fancy patch management tools, a handful of Windows systems on every network I come across don't have all the patches applied. Even if you think all your systems have the latest patches installed, you have to be sure. It's what ethical hacking is all about: trust but verify.

Before you go 'sploitin' vulnerabilities with Metasploit, it's very important to know that you're venturing into sensitive territory. Not only can you gain full, unauthorized access to sensitive systems, you can also put your test systems into a state where they can lock up or reboot. So, read each exploit's documentation and proceed with caution.

Before you can seriously exploit a missing patch or related vulnerability, you have to find out what's exploitable. The best way to go about doing this is to use a tool such as QualysGuard or LANguard to find them. I've found QualysGuard to be very good at rooting out such vulnerabilities even as an unauthenticated user on the network. Figure 10-11 shows QualysGuard scan results of a Windows server system that has the nasty Windows Plug and Play Remote Code Execution vulnerability.

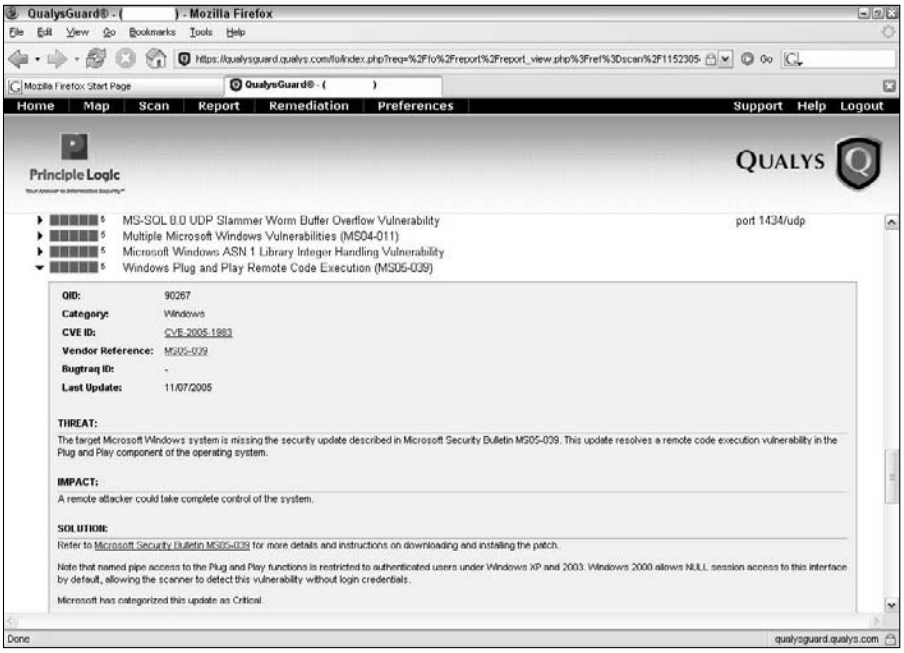


Figure 10-11: Exploitable vulnerability found by Qualys Guard.

Using Metasploit

After you find a vulnerability, the next step is to exploit it. In this example, I use Metasploit (an open source tool now owned by Rapid7) and obtain a remote command prompt on the vulnerable server. Here's how:

1. **Download and install Metasploit from `www.metasploit.com/framework`.**

   I use the Windows version; all you have to do is download and run the executable. The process takes a couple of minutes because it has to install the Linux/UNIX environment, called cygwin, for Windows. There's a version of Metasploit for Linux/UNIX, too.

2. **After the installation is complete, run the Metasploit GUI, which is Metasploit's main console.**

   There's also a Web-based version of Metasploit that you can access through your browser (Metasploit Web) but I prefer the GUI interface.

   You see a screen similar to the one shown in Figure 10-12.

3. **Expand the Exploits option to see what exploits are available to run, as shown in Figure 10-13.**

   If you know the specific vulnerability (say Microsoft's MS08-067), you can simply enter part or all of the search term (such as **ms08**) in the search field at the top and then click Find.

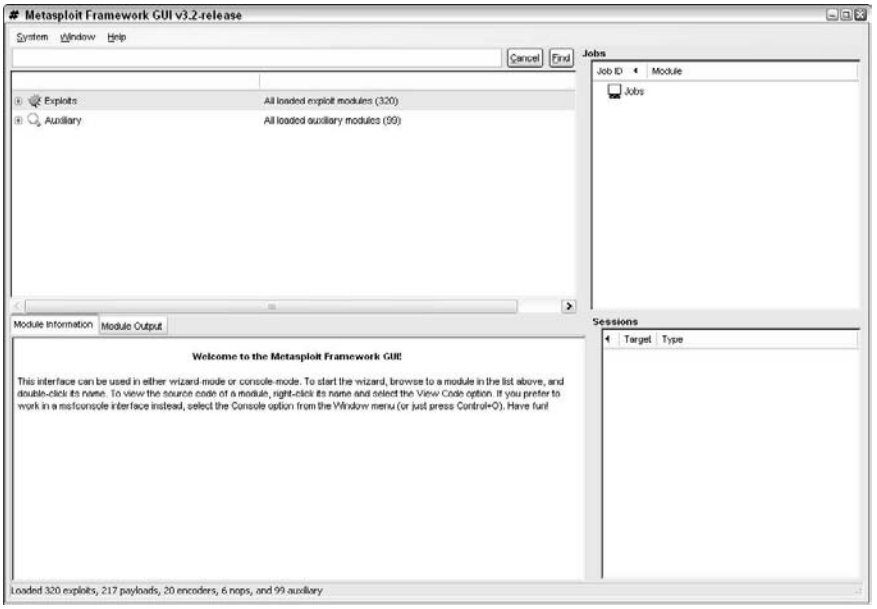


Figure 10-12: Main Metasploit console.

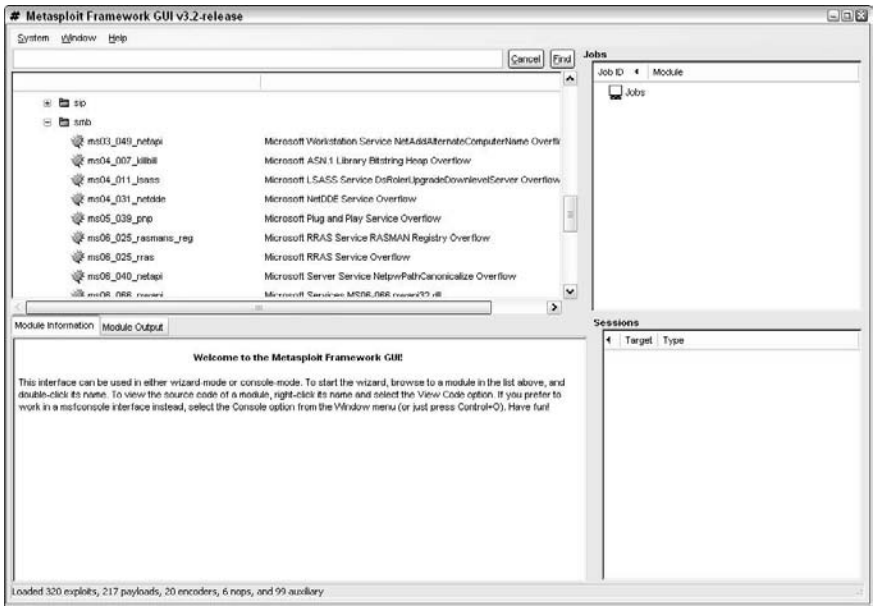


Figure 10-13: Browsing the available exploits.

4. **After you find the exploit you wish to run against your target system, simply double-click the exploit and then follow the steps starting with selecting the target operating system, as shown in Figure 10-14; click Forward.**

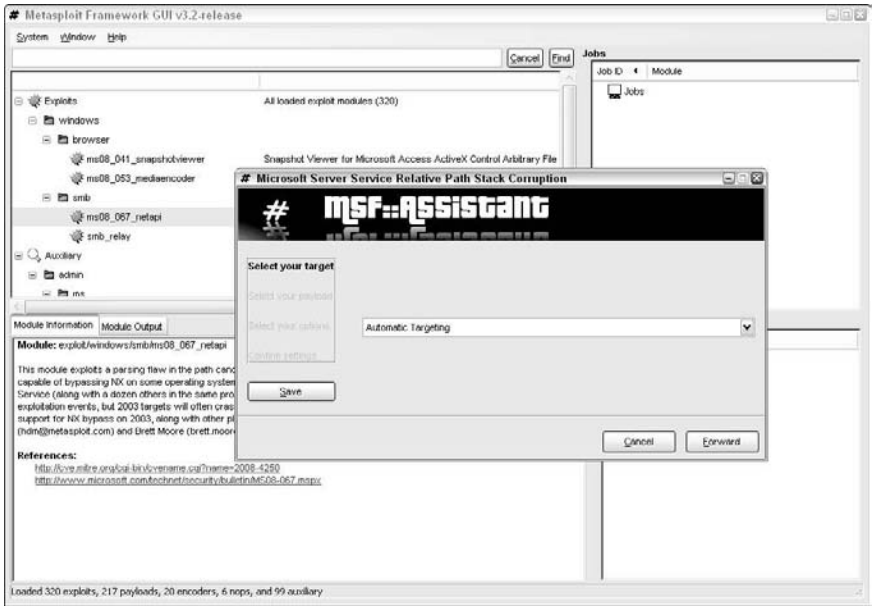


Figure 10-14: Select the target operating system.

Select Automatic Targeting if it's available; otherwise, make your best guess of which version of Windows is running and then click Forward.

5. **Select the payload (the specific hack) you wish to send to the target, and click Forward.**

   I typically choose windows/shell/reverse_tcp, as shown in Figure 10-15.

Figure 10-15: Load a specific payload to send to the exploited system.

6. **Enter the IP address of the target system in the RHOST field and confirm that the IP address shown in the LHOST field is the address of your testing system, as shown in Figure 10-16, and click Forward.**

7. **Confirm your settings on the final screen, as shown in Figure 10-17, and click Apply.**

   The job executes, and you see the shell session in the Sessions section in the lower-right quadrant of the Metasploit GUI.

Figure 10-16: Entering required remote and IP address.

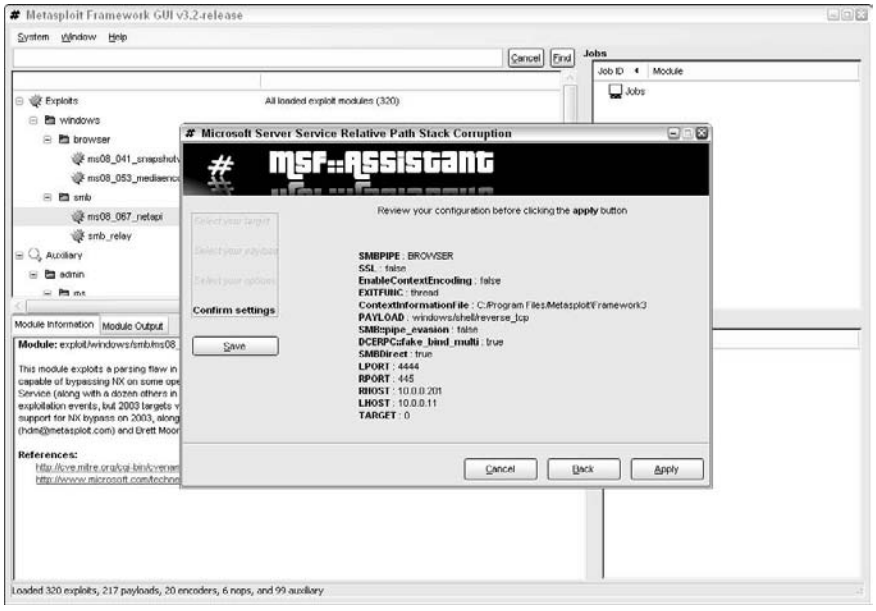


Figure 10-17: Checking final parameters before carrying out the exploit.

8. **Double-click the session and a new window opens with a command prompt on the target system, as shown in Figure 10-18.**

   I now "own" the system and can do whatever I want.

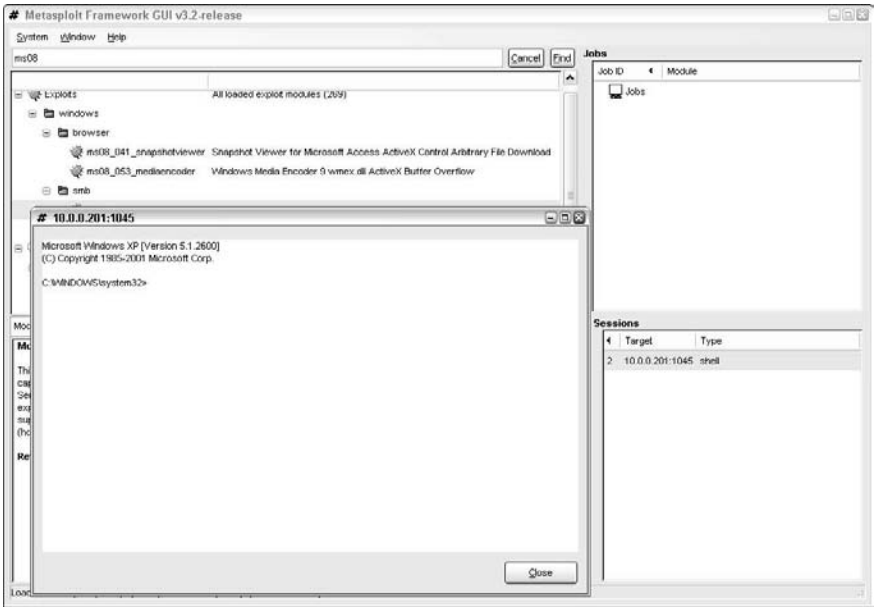


Figure 10-18: Remote command prompt on target system obtained by exploiting a missing patch vulnerability.

For example, one thing I commonly do is add a user account to the exploited system. You can actually do this within Metasploit (via the adduser payloads) but I prefer to do it on my own so I can get screenshots of my actions. To add a user, simply enter **net user username password /add** at the Metapsloit command prompt.

Next, I add the user to the local administrators group by entering **net localgroup administrators username /add** at the Metasploit command prompt. You can then log in to the remote system by mapping a drive to the C$ share or by connecting via Remote Desktop.

If you choose to add a user account during this phase, be sure to remove it when you finish. Otherwise, you can create another vulnerability on the system — especially if the account has a weak password.

All in all, this is ethical hacking at its finest!

Keep in mind that I demonstrate only a small fraction of what Metasploit can do. I highly recommend you download it and familiarize yourself with it. Numerous resources on Metasploit include the Metasploit mailing list, which can be found at `www.metasploit.com/framework/support`. The power of Metasploit is unbelievable — especially when combined with the exploit code that's continually updated at the milw0rm site (`www.milw0rm.com`).

Countermeasures against missing patch vulnerability exploits

Patch your systems. Seriously, that's all there is to it. Combine that with the other hardening recommendations I provide in this chapter, and you have a pretty darned secure Windows environment.

To get your arms around the patching process, you have to automate it wherever you can. You can use Windows Update — or better yet — Windows Server Update Services (WSUS), which can be found at `http://technet.microsoft.com/en-us/wsus/default.aspx`. If you're looking for a commercial alternative such as BigFix Patch Management (`www.bigfix.com/content/patch-management`) and Lumension Patch and Remediation (`www.lumension.com/vulnerability-management/patch-management-software.jsp`).

Authenticated Scans

Another test you can run against your Windows systems is an "authenticated" scan — essentially looking for vulnerabilities as a trusted insider. I find these types of tests to be very beneficial because they often highlight system problems and even operational security weaknesses (such as poor change management processes and lack of information classification) that would never be discovered otherwise.

A trusted insider who has physical access to your network and the right tools can exploit vulnerabilities even more easily. This is especially true if no internal access control lists or IPS is in place.

A way to look for Windows weaknesses while you're logged in (that is, through the eyes of a malicious insider) is by using some of the general vulnerability scanning tools I've mentioned, such as LANguard and QualysGuard. Figure 10-19 shows confirmed and potential security issues found on a Windows 7 system.

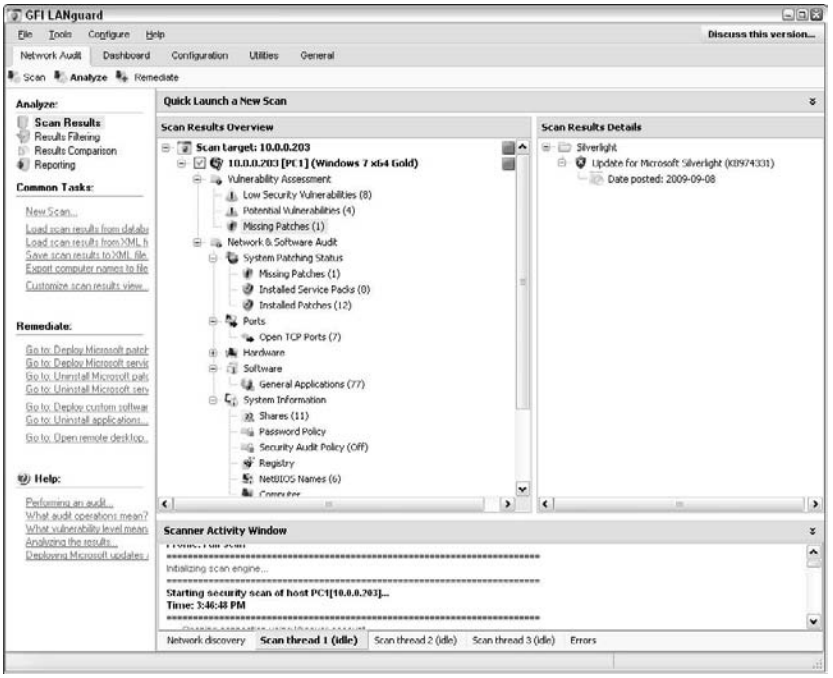


Figure 10-19: Running an authenticated scan with LANguard to see what rogue insiders can exploit.

I recommend running authenticated scans as a regular local or domain user and as an administrator or any other user type you might have. This will show you who has access to what in the event a vulnerability is present. You'll likely be surprised to find out that a large portion of vulnerabilities, such as those listed in Figure 10-19, are accessible via a standard user account.

You can also use Microsoft Baseline Security Analyzer (MBSA) to check for basic vulnerabilities and missing patches. MBSA is a free utility from Microsoft that can be downloaded at `www.microsoft.com/technet/security/tools/mbsahome.mspx`. MBSA checks all Windows 2000 and later operating systems for missing patches. It also tests Windows, SQL Server, and IIS for basic security settings, such as weak passwords. You can use these tests to identify security weaknesses in your systems.

With MBSA, you can scan either the local system you're logged into or computers across the network. One caveat: MBSA requires an administrator account on the local machines you're scanning.

Chapter 11

Linux

In This Chapter

- Examining Linux hacking tools
- Port-scanning a Linux server
- Gleaning Linux information without logging in
- Exploiting common vulnerabilities when logged in to Linux
- Minimizing Linux security risks

Linux — the darling competitor to Microsoft — is the latest flavor of UNIX to take off in corporate networks. A common misconception is that the majority of security vulnerabilities are in the Windows operating system (OS). However, security experts see more and more that Linux and its sister variants of UNIX are prone to some of the same types of security vulnerabilities.

Hackers are attacking Linux in droves because of its popularity and growing usage in today's network environment. Because some versions of Linux are *free* — in the sense that you don't have to pay for the base operating system — many organizations are installing Linux for their Web servers and e-mail servers in hopes of saving money and having a more secure system. Linux has grown in popularity for other reasons as well, including the following:

- Abundant resources are available, including books, Web sites, and developer and consultant expertise.
- Unlikeliness that Linux will be hit with as much malware as Windows and its applications do. Linux excels when it comes to security, but it probably won't stay that way.
- Increased buy-in from other UNIX vendors, including IBM and Sun Microsystems. Even Novell stopped development of its mighty NetWare OS and is now focusing on a Linux-based kernel.
- Growing ease of use.

Based on what I see in my work, Linux is less vulnerable to common security flaws than Windows. When comparing any current distribution of Linux, such as Ubuntu and Red Hat/Fedora, with Windows XP, Windows Vista, or

Windows 7, I tend to find a lot more weaknesses in Windows systems. Chalk it up to widespread use, more features, or uneducated users, but there seems to be a lot more that can happen in a Windows environment. That said, Linux is certainly not flawless. In addition to the password attacks I cover in Chapter 7, certain remote and local attacks are possible against Linux-based systems. In this chapter, I show you some security issues in the Linux operating system and outline some countermeasures to plug the holes so you can keep the bad guys out. Don't let the title of this chapter fool you — a lot of this information applies to all flavors of UNIX.

Linux Vulnerabilities

Vulnerabilities and attacks against Linux are creating business risks in a growing number of organizations — especially e-commerce companies, network product vendors, and ISPs that rely on Linux for many of their systems. When Linux systems are hacked, the victim organizations can experience the same side effects as their Windows-using counterparts, including:

- Leakage of sensitive information
- Cracked passwords
- Corrupted or deleted databases
- Systems taken completely offline

Choosing Tools

You can use many UNIX-based security tools to test your Linux systems. Some are much better than others. I often find that my Windows-based commercial tools do as good a job as any. My favorites are as follows:

- Windows-based **SuperScan** version 3 (`www.foundstone.com/resources/proddesc/superscan3.htm`) for ping sweeps and TCP port scanning
- **Nmap** (`http://nmap.org`) for OS fingerprinting and more detailed port scanning
- Windows-based **LANguard** (`www.gfi.com/lannetscan`) for port scanning, OS enumeration, and vulnerability testing
- **THC-Amap** (`http://freeworld.thc.org/thc-amap`) for application version mapping
- **Tiger** (`ftp://ftp.debian.org/debian/pool/main/t/tiger`) for automatically assessing local system security settings

- **Linux Security Auditing Tool (LSAT)** (`http://usat.sourceforge.net`) for automatically assessing local system security settings
- **QualysGuard** (`www.qualys.com`) for OS fingerprinting, port scanning, and very detailed and accurate vulnerability testing
- **Nessus** (`www.nessus.org`) for OS fingerprinting, port scanning, and vulnerability testing
- **BackTrack** (`www.remote-exploit.org/backtrack.html`) toolset on a bootable CD or `.iso` image file

Hundreds if not thousands of other Linux hacking and testing tools are available on such sites as SourceForge.net (`http://sourceforge.net`) and freshmeat.net (`http://freshmeat.net`). The key is to find a set of tools — preferably as few as possible — that can do the job that you need to do and that you feel comfortable working with.

Information Gathering

You can scan your Linux-based systems and gather information from both outside (if the system is a publicly accessible host) and inside your network.

Scan from both directions so you see what the bad guys can see from outside and inside the network.

System scanning

Linux services — called *daemons* — are the programs that run on a system and serve up various services and applications for users.

- Internet services, such as the Apache Web server (httpd), telnet (telnetd), and FTP (ftpd), often give away too much information about the system, including software versions, internal IP addresses, and usernames. This information can allow hackers to exploit a known weakness in the system.
- TCP and UDP *small services,* such as echo, daytime, and chargen, are often enabled by default and don't need to be.

The vulnerabilities inherent in your Linux systems depend on what services are running. You can perform basic port scans to glean information about what's running.

The SuperScan results in Figure 11-1 show many potentially vulnerable services on this Linux system, including remote procedure call (RPC), a Web server, telnet, and FTP.

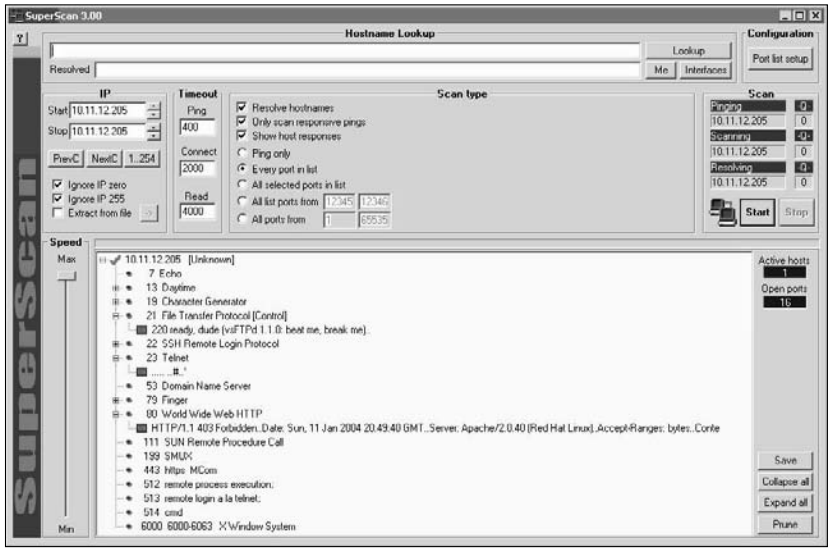


Figure 11-1: Port-scanning a Linux server with SuperScan.

In addition to SuperScan, you can run another scanner, such as Nessus or LANguard Network Security Scanner, against the system to try to gather more information, including:

- A vulnerable version of OpenSSH, as shown in Figure 11-2
- The finger service information returned by LANguard Network Security Scanner, as shown in Figure 11-3

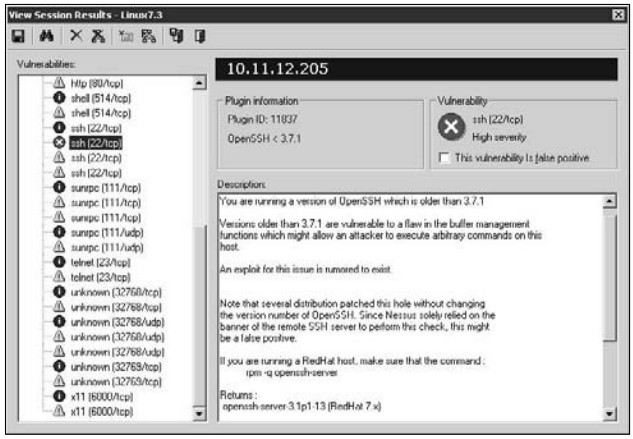


Figure 11-2: Using Nessus to discover a vulnerability with OpenSSH.

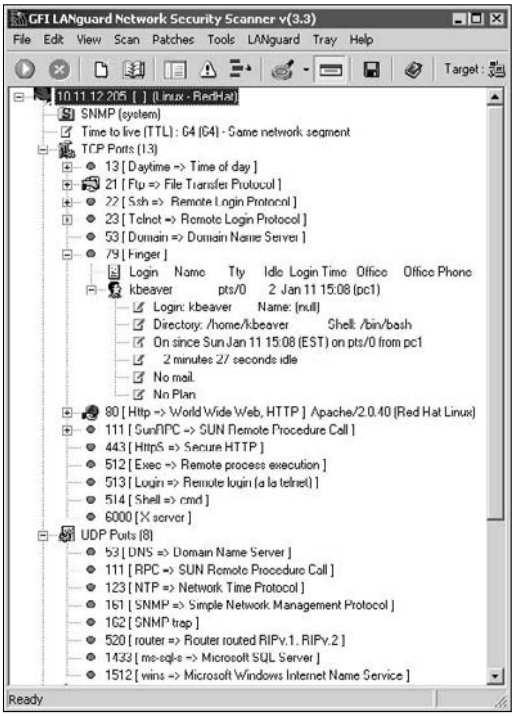


Figure 11-3: LANguard revealing user information via the finger service.

LANguard also determined that the server is running rlogin and rexec, the Berkeley Software Distribution (BSD) r-services. Figure 11-3 also shows that LANguard thinks the remote operating system is Red Hat Linux. This information can be handy when you come across unfamiliar open ports.

Figure 11-4 shows various r-services and other daemons that network administrators are notorious for leaving running unnecessarily on UNIX-based operating systems. Notice that LANguard points out specific vulnerabilities associated with some of these services, along with a recommendation to use SSH as an alternative.

You can go a step further and find out the exact distribution and kernel version by running an OS fingerprint scan with Nmap, as shown in Figure 11-5.

The Windows-based NetScanTools Pro also has the ability to determine the version of Linux that's running, as shown in Figure 11-6.

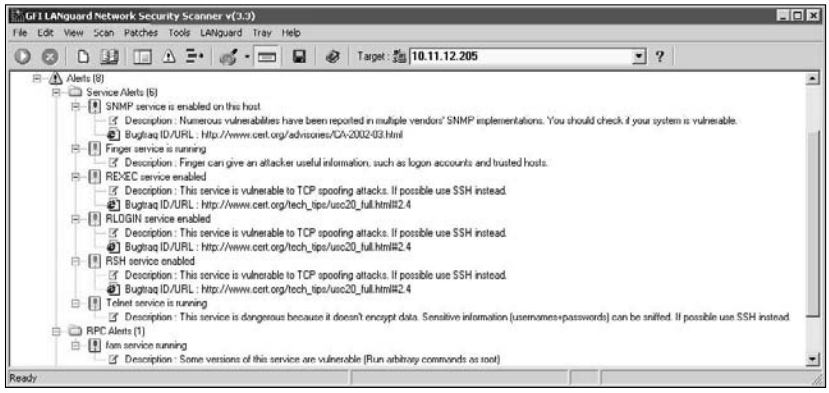


Figure 11-4: Potentially vulnerable r-services found by LANguard.

Figure 11-5: Using Nmap to determine the OS kernel version of a Linux server.

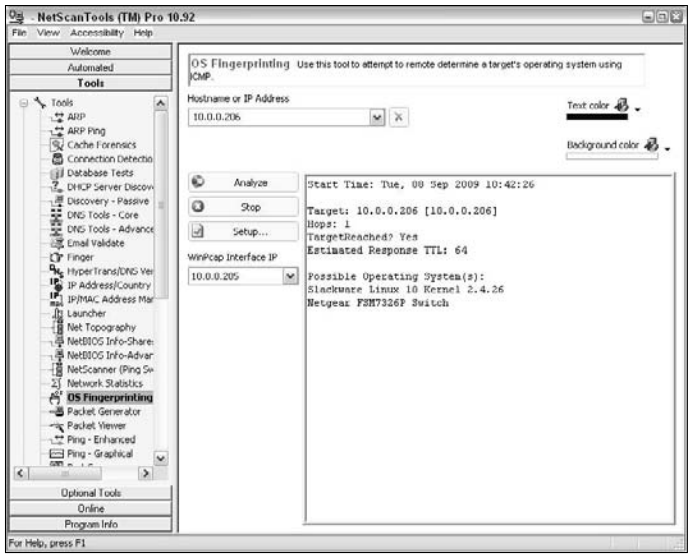


Figure 11-6: Using NetScan Tools Pro to determine that Slackware Linux is running.

Countermeasures against system scanning

Although you can't completely prevent system scanning, you can still implement the following countermeasures to keep the bad guys from gleaning too much information about your systems:

- Protect the systems with either
  - A firewall, such as netfilter/iptables (`www.netfilter.org`)
  - A host-based intrusion-prevention application, such as PortSentry (`http://sourceforge.net/projects/sentrytools`) and SNARE (`www.intersectalliance.com/projects/Snare`).
- Disable the services you don't need, including RPC, HTTP, FTP, and telnet. You might very well need some of these services — just make sure you have a business need for them. This keeps the services from showing up in a port scan, which gives an attacker less incentive to break in to your system.
- Make sure the latest software and patches are loaded to reduce the chance of exploitation if an attacker determines what services you're running.

Unneeded and Unsecured Services

When you know which daemons and applications are running — such as FTP, telnet, and a Web server — it's nice to know exactly which versions are running so you can look up their associated vulnerabilities and decide whether to turn them off. The National Vulnerability Database site (`http://nvd.nist.gov`) is a good resource for determining vulnerabilities.

Searches

Several security tools can help determine vulnerabilities. These types of utilities might not identify all applications down to the exact version number, but they're a very powerful way of collecting system information.

Vulnerabilities

Be especially mindful of these known security weaknesses in a system:

- FTP — especially if it isn't properly configured — can provide a way for an attacker to download and access files on your system.
- Telnet and FTP are vulnerable to network analyzer captures of the cleartext user ID and password the applications use. Their logins can also be brute-force attacked.

- Old versions of sendmail — the world's most popular e-mail server — have many security issues.

  Make sure sendmail is patched and hardened.

- R-services, such as rlogin, rdist, rexecd, rsh, and rcp, are especially vulnerable to attacks.

Many Web servers run on Linux so you can't overlook the importance of checking for weaknesses in Apache, Tomcat, and your specific applications. For example, a common Linux vulnerability is that user names can be determined via Apache when it doesn't have the UserDir directive disabled in its `httpd.conf` file. You can exploit this weakness manually by browsing to well-known user folders, such as *`http://www.your~site.com/user_name`* or, better yet, by using a tool, such as WebInspect or QualysGuard, to automatically enumerate the system. Either way, you can find out which Linux users exist and then launch a Web password-cracking attack. There are also numerous ways to access system files (including `/etc/passwd`) via vulnerable CGI code. I cover hacking Web applications in Chapter 14.

Likewise, FTP is often running unsecured on Linux systems. I've found Linux systems with anonymous FTP enabled that were sharing sensitive healthcare and financial information to everyone on the local network. Talk about a lack of accountability! So, don't forget to look for the simple stuff. When hacking Linux, you can dig down deep into the kernel and do this and that to exploit the system but it's usually the little things that get you.

Tools

The following tools can perform more in-depth information gathering beyond port scanning to enumerate your Linux systems and see what the hackers see:

- Nmap can check for specific versions of the services loaded, as shown in Figure 11-7. Simply run Nmap with the `-sV` command-line switch.

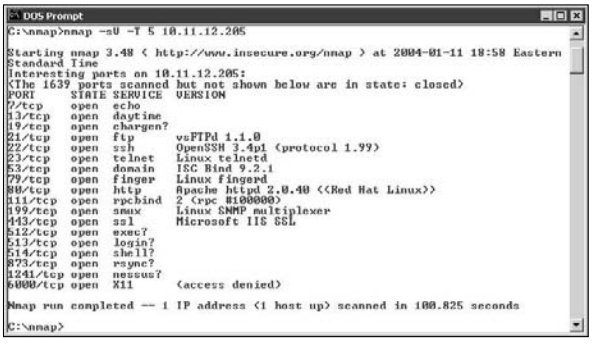


Figure 11-7: Using Nmap to check application versions.

- Amap is similar to Nmap, but it has a couple of advantages:
  - Amap is much faster for these types of scans.
  - Amap can detect applications that are configured to run on non-standard ports, such as Apache running on port 6789 instead of its default 80.

  The output of an Amap scan of the localhost (hence, the 127.0.0.1 address) is shown in Figure 11-8. Amap was run with the following options to enumerate some commonly hacked ports:
  - `-1` makes the scan run faster.
  - `-b` prints the responses in ASCII characters.
  - `-q` skips reporting of closed ports.
  - `21` probes the FTP control port.
  - `22` probes the SSH port.
  - `23` probes the telnet port.
  - `80` probes the HTTP port.

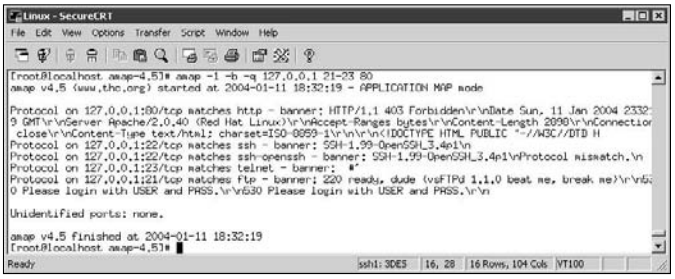


Figure 11-8: Using Amap to check application versions.

- `netstat` shows the services running on a local machine. Enter this command while logged in:

```
netstat -anp
```

- List Open Files (`lsof`) displays processes that are listening and files that are open on the system.

  To run `lsof`, login and enter this command at a Linux command prompt: `lsof -i +M`. `lsof` can come in handy when you suspect that malware has found its way onto the system.

Countermeasures against attacks on unneeded services

You can and should disable the unneeded services on your Linux systems. This is one of the best ways to keep your Linux system secure. Like reducing the number of entry points (such as open doors and windows) in your house, the more entry points you eliminate, the fewer places an intruder can break in.

Disabling unneeded services

The best method of disabling unneeded services depends on how the daemon is loaded in the first place. You have several places to disable services, depending on the version of Linux you're running.

If you don't need to run a particular service, take the safe route: Turn it off!

inetd.conf

If it makes good business sense — that is, if you don't need them — disable unneeded services by commenting out the loading of daemons you don't use. Follow these steps:

1. **Enter the following command at the Linux prompt:**

   ```
   ps -aux
   ```

   The process ID (PID) for each daemon, including inetd, is listed on the screen. In Figure 11-9, the PID for the sshd (Secure Shell daemon) is 646.

2. **Copy the PID for inetd from the screen on a piece of paper.**

3. **Open `/etc/inetd.conf` in the Linux text editor vi by entering the following command:**

   ```
   vi /etc/inetd.conf
   ```

4. **When you have the file loaded in vi, enable the insert (edit) mode by pressing I.**

5. **Move the cursor to the beginning of the line of the daemon that you want to disable, such as httpd (Web server daemon) and type # at the beginning of the line.**

   This comments out the line and prevents it from loading when you reboot the server or restart inetd.

6. **To exit vi and save your changes, press Esc to exit the insert mode, type** :wq**, and then press Enter.**

   This tells vi that you want to write your changes and quit.

7. **Restart inetd by entering this command with the inetd PID:**

```
kill –HUP PID
```

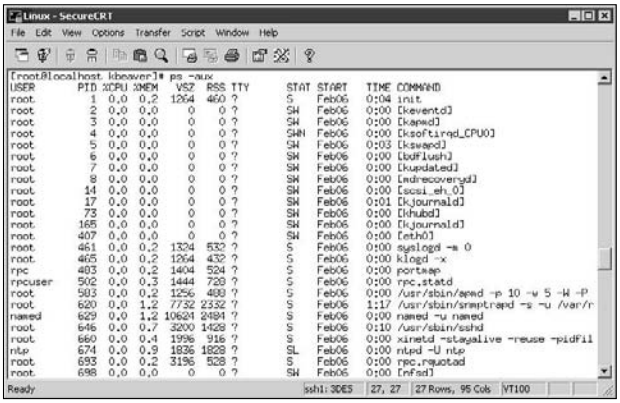


Figure 11-9: Viewing the process IDs for running daemons by using ps -aux.

chkconfig

If you don't have an `inetd.conf` file (or it's empty), your version of Linux is probably running the xinetd program (`www.xinetd.org`) — a more secure replacement for inetd — to listen for incoming network application requests. You can edit the `/etc/xinetd.conf` file if this is the case. For more information on the usage of xinetd and `xinetd.conf`, enter **man xinetd** or **man xinetd.conf** at a Linux command prompt. If you're running Red Hat 7.0 or later, you can run the `/sbin/chkconfig` program to turn off the daemons you don't want to load.

For example, you can enter the following to disable the snmp daemon:

```
chkconfig --del snmpd
```

You can also enter **chkconfig –list** at a command prompt to see what services are enabled in the `xinetd.conf` file.

You can use the chkconfig program to disable other services, such as FTP, telnet, and Web server.

Access control

TCP Wrappers can control access to critical services that you run, such as FTP or HTTP. This program controls access for TCP services and logs their usage, helping you control access via hostname or IP address and track malicious activities.

You can download TCP Wrappers from `http://itso.iu.edu/TCP_Wrappers`.

Always make sure that your operating system and the applications running on it are not open to the world (or your internal network) by ensuring that reasonable password requirements are in place. Don't forget to disable anonymous FTP unless you absolutely need it. Even if you do, limit system access to only those with a business need to access sensitive information.

.rhosts and hosts.equiv Files

Linux — and all the flavors of UNIX — are file-based operating systems. Practically everything that's done on the system involves the manipulation of files. This is why so many attacks against Linux are at the file level.

Hacks using the .rhosts and hosts.equiv files

If hackers can capture a user ID and password by using a network analyzer or can crash an application and gain root access via a buffer overflow, one thing they look for is what users are trusted by the local system. That's why it's critical to assess these files yourself. The `/etc/hosts.equiv` and `.rhosts` files list this information.

hosts.equiv

The `/etc/hosts.equiv` file won't give away root access information, but it does specify which accounts on the system can access services on the local host. For example, if *tribe* were listed in this file, all users on the tribe system would be allowed access. As with the `.rhosts` file, external hackers can read this file and then spoof their IP address and hostname to gain unauthorized access to the local system. Hackers can also use the names located in the `.rhosts` and `hosts.equiv` files to look for names of other computers to attack.

.rhosts

The `$home/.rhosts` files in Linux specify which remote users can access the Berkeley Software Distribution (BSD) r-commands (such as rsh, rcp, and rlogin) on the local system without a password. This file is in a specific user's (including root) home directory, such as `/home/jsmith`. An `.rhosts` file may look like this:

```
tribe   scott
tribe   eddie
```

This file allows users Scott and Eddie on the remote-system tribe to log in to the local host with the same privileges as the local user. If a plus sign (+) is entered in the remote-host and user fields, any user from any host could log in to the local system. The hacker can add entries into this file by:

- Manually manipulating the file.
- Running a script that exploits an unsecured Common Gateway Interface (CGI) script on a Web-server application that's running on the system.

This configuration file is a prime target for a malicious attack. On most Linux systems I've tested, these files aren't enabled by default. However, a user can create one in his or her home directory on the system — intentionally or accidentally — which can create a major security hole on the system.

Countermeasures against .rhosts and hosts.equiv file attacks

Use both of the following countermeasures to prevent hacker attacks against the `.rhosts` and `hosts.equiv` files in your Linux system.

Disabling commands

A good way to prevent abuse of these files is to disable the BSD r-commands. This can be done in two ways:

- Comment out the lines starting with `shell`, `login`, and `exec` in `inetd.conf`.
- Edit the `rexec`, `rlogin`, and `rsh` files located in the `/etc/xinetd.d` directory. Open each file in a text editor and change `disable=no` to `disable=yes`, as shown in Figure 11-10.

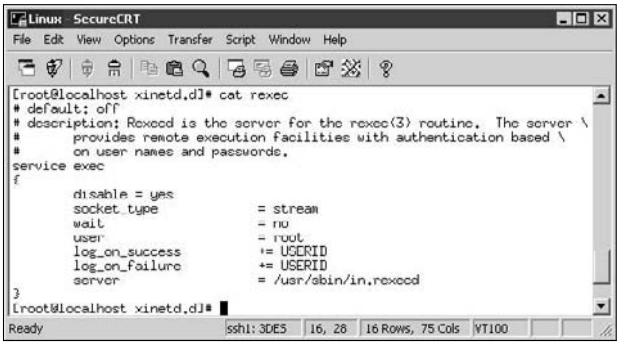


Figure 11-10: The rexec file showing the disable option.

In Red Hat Enterprise Linux, you can disable the BSD r-commands with the setup program:

1. **Enter** setup **at a command prompt.**
2. **Choose System Services from the menu.**
3. **Remove the asterisks next to each of the r-services.**

Blocking access

A couple of countermeasures can block rogue access of the `.rhosts` and `hosts.equiv` files:

- Block spoofed addresses at the firewall, as I outline in Chapter 8.
- Set the read permissions for each file's owner only.
  - `.rhosts`: Enter this command in each user's home directory:

```
chmod 600 .rhosts
```

  - `hosts.equiv`: Enter this command in the `/etc` directory:

```
chmod 600 hosts.equiv
```

You can also use Tripwire (`http://sourceforge.net/projects/tripwire/`) to monitor these files and alert you when access is obtained or changes are made.

NFS

The Network File System (NFS) is used to mount remote file systems (similar to shares in Windows) from the local machine. Given the remote access nature of NFS, it certainly has its fair share of hacks. I cover additional storage vulnerabilities and hacks in Chapter 15.

NFS hacks

If NFS was set up improperly or its configuration has been tampered with — namely, the `/etc/exports` file containing a setting that allows the world to read the entire file system — remote hackers can easily obtain remote access and do anything they want on the system. All it takes is a line, such as the following, in the `/etc/exports` file:

```
/   rw
```

This line says that anyone can remotely mount the root partition in a read-write fashion. Of course, the following conditions must also be true:

- The NFS daemon (nfsd) must be loaded, along with the portmap daemon that would map NFS to RPC.
- The firewall must allow the NFS traffic through.
- The remote systems that are allowed into the server running the NFS daemon must be placed into the `/etc/hosts.allow` file.

This remote-mounting capability is easy to misconfigure. It's often related to a Linux administrator's misunderstanding of what it takes to share out the NFS mounts and resorting to the easiest way possible to get it working. After hackers gain remote access, the system is theirs.

Countermeasures against NFS attacks

The best defense against NFS hacking depends on whether you actually need the service running.

- If you don't need NFS, disable it.
- If you need NFS, implement both of the following countermeasures:
  - Filter NFS traffic at the firewall — typically, TCP port 111 (the port-mapper port) if you want to filter all RPC traffic.
  - Make sure that your `/etc/exports` and `/etc/hosts.allow` files are configured properly to keep the world outside your network.

File Permissions

In Linux, special file types allow programs to run with the file owner's rights:

- SetUID (for user IDs)
- SetGID (for group IDs)

SetUID and SetGID are required when a user runs a program that needs full access to the system to perform its tasks. For example, when a user invokes the passwd program to change his or her password, the program is actually loaded and run without root or any other user's privileges. This is done so that the user can run the program and the program can update the password database without the root account being involved in the process.

File permission hacks

By default, rogue programs that run with root privileges can be easily hidden. An external attacker or malicious insider might do this to hide hacking files, such as rootkits, on the system. This can be done with SetUID and SetGID coding in their hacking programs.

Countermeasures against file permission attacks

You can test for rogue programs by using both manual and automated testing methods.

Manual testing

The following commands can identify and print to the screen SetUID and SetGID programs:

- Programs that are configured for SetUID:

```
find / -perm -4000 -print
```

- Programs that are configured for SetGID:

```
find / -perm -2000 -print
```

- Files that are readable by anyone in the world:

```
find / -perm -2 -type f -print
```

- Hidden files:

```
find / -name ".*"
```

You probably have hundreds of files in each of these categories, so don't be alarmed. When you discover files with these attributes set, you need to make sure that they are actually supposed to have those attributes by researching in your documentation or on the Internet, or by comparing them to a known secure system or data backup.

Keep an eye on your systems to detect any new SetUID or SetGID files that suddenly appear.

Automatic testing

You can use an automated file-modification auditing program to alert you when these types of changes are made. This is what I recommend — it's a lot easier on an ongoing basis.

- A change-detection application, such as Tripwire, can help you keep track of what changed and when.
- A file-monitoring program, such as COPS (`ftp://ftp.cerias.purdue.edu/pub/tools/unix/scanners/cops`), finds files that have changed in status (such as a new SetUID or removed SetGID).

Buffer Overflows

RPC and other vulnerable daemons are common targets for buffer-overflow attacks. Buffer-overflow attacks are often how the hacker can get in to modify system files, read database files, and more.

Attacks

In a buffer-overflow attack, the attacker either manually sends strings of information to the victim Linux machine or writes a script to do so. These strings contain

- Instructions to the processor to basically do nothing.
- Malicious code to replace the attacked process.

  For example, `exec ("/bin/sh")` creates a shell command prompt.
- A pointer to the start of the malicious code in the memory buffer.

If an attacked application (such as FTP or RPC) is running as root (certain programs do), this can give attackers root permissions in their remote shells. Specific examples of vulnerable software running on Linux are Samba, MySQL, and Firefox. Depending on the version, this software can be exploited using the free Metasploit tool (`www.metasploit.com`) to obtain remote command prompts, add backdoor user accounts, change ownership of files, and more. I cover Metasploit in Chapter 10.

Countermeasures against buffer-overflow attacks

Three main countermeasures can help prevent buffer-overflow attacks:

- Disable unneeded services.
- Protect your Linux systems with either a firewall or a host-based intrusion prevention.
- Enable another access control mechanism, such as TCP Wrappers, that authenticates users with a password.

  Don't just enable access controls via an IP address or hostname. That can easily be spoofed.

As always, make sure that your systems have been updated with the latest kernel and security patches.

Physical Security

Some Linux vulnerabilities involve the bad guy actually being at the system console — something that's entirely possible given the insider threats that every organization faces.

Physical security hacks

When a hacker is at the system console, anything goes, including rebooting the system (even if no one is logged in) by pressing Ctrl+Alt+Delete. After the system is rebooted, the hacker can start it in single-user mode, which allows the hacker to zero out the root password or possibly even read the entire shadow password file. I cover Linux password cracking in Chapter 7.

Countermeasures against physical security attacks

Edit your `/etc/inittab` file and comment out (place a # sign in front of) the line that reads `ca::ctrlaltdel:/sbin/shutdown -t3 -r now`, shown in the last line of Figure 11-11. This will prevent someone from rebooting the system by pressing Ctrl+Alt+Delete. Be forewarned that this will also prevent you from legitimately using Ctrl+Alt+Delete.

For Linux-based laptops, using disk encryption software such as TrueCrypt (`http://www.truecrypt.org`) is a must. If you don't, when a laptop is lost or stolen, you could very well have a data breach on your hands and all the state federal compliance requirements that go along with it. Not good!

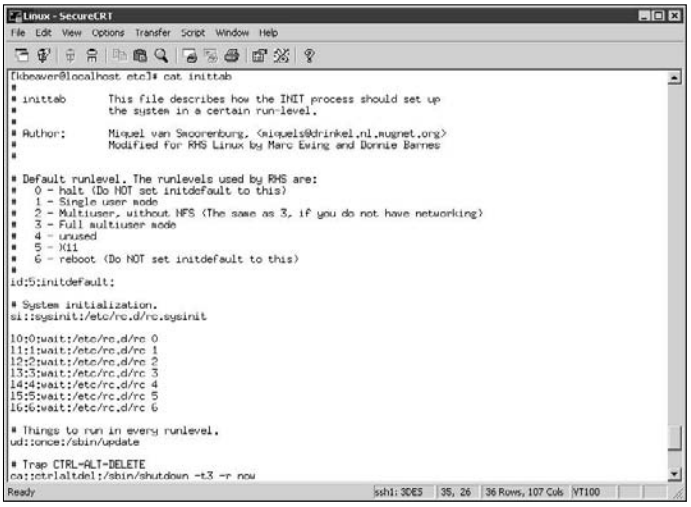


Figure 11-11: /etc/inittab showing the line that allows a Ctrl+Alt+Delete shutdown.

If you believe that someone has recently gained access to your system, either physically or by exploiting a vulnerability, such as a weak password or buffer overflow, you can use last, the program, to view the last few logins into the system to check for strange login IDs or login times. This program peruses the `/var/log/wtmp` file and displays the users who logged in last. You can enter `last | head` to view the first part of the file (the first ten lines) if you want to see the most recent logins.

General Security Tests

You can assess critical, and often overlooked, security issues on your Linux systems, such as the following:

- Misconfigurations or unauthorized entries in the shadow password files
- Password complexity requirements
- Users equivalent to root
- Suspicious automated tasks configured in cron, the script scheduler program
- Signature checks on system binary files
- Checks for rootkits
- Network configuration, including measures to prevent packet spoofing and other denial of service (DoS) attacks
- Permissions on system log files

You can do all these assessments manually — or better yet, use an automated tool to do it for you! Figure 11-12 shows the initiation of the Tiger security-auditing tool (`www.nongnu.org/tiger`), and Figure 11-13 shows a portion of the audit results. Talk about some great bang for no buck with this tool!

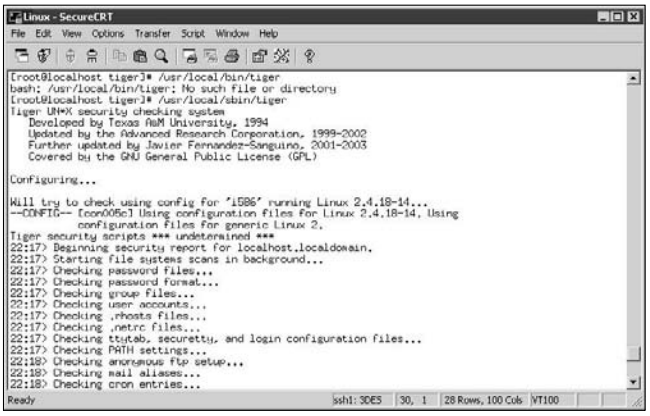


Figure 11-12: Running the Tiger security-auditing tool.

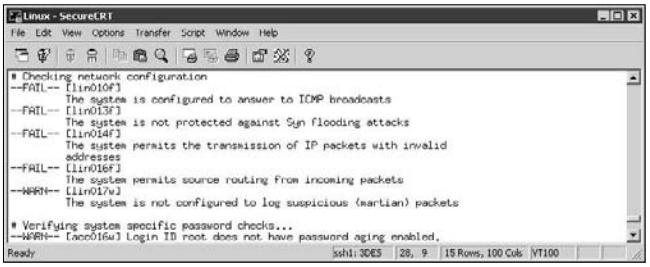


Figure 11-13: Partial output of the Tiger tool.

Alternatives to Tiger include Linux Security Auditing Tool (LSAT; `http://usat.sourceforge.net`) as well as the Bastille Hardening program (`http://bastille-linux.sourceforge.net`).

Patching Linux

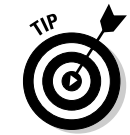
Ongoing patching is perhaps the best thing you can do to enhance the security of your Linux systems. Regardless of the Linux distribution you use, using a tool to assist in your patching efforts makes your job a lot easier.

I often find Linux is completely out of the patch management loop. With the focus on patching Windows, many network administrators forget about the Linux systems they have on their network. Don't fall into this trap.

Distribution updates

The distribution process is different on every distribution of Linux. You can use the following tools, based on your specific distribution:

- **Red Hat:** The following tools update Red Hat/Fedora Linux systems:
  - Red Hat Package Manager (RPM), which is the GUI-based application that runs in the Red Hat GUI desktop. It manages files with an `.rpm` extension that Red Hat and other freeware and open source developers use to package their programs.
  - up2date, a command-line, text-based tool that's included in Red Hat/Fedora.
- **Debian:** You can use the Debian Package System (dpkg) included with the operating system to update Debian Linux systems.
- **Slackware:** You can use the Slackware Package Tool (pkgtool) included with the operating system to update Slackware Linux systems.
- **SUSE:** SUSE Linux includes the YaST2 Package Manager.



In addition to Linux kernel and general operating system updates, make sure you pay attention to Apache, OpenSSL, OpenSSH, MySQL, and other software on your systems. They have weaknesses that you probably don't want to overlook.

Multiplatform update managers

The open source option for multiple Linux platforms called RPM Package Manager (`www.rpm.org`) is worth checking out. Commercial tools have additional features, such as correlating patches with vulnerabilities and automatically deploying appropriate patches. Commercial tools that can help with Linux patch management include BigFix Patch Management (`www.bigfix.com/content/patch-management`) and Lumension Patch and Remediation (`www.lumension.com/vulnerability-management/patch-management-software.jsp`).

Chapter 12

Novell NetWare

In This Chapter

- Selecting NetWare hacking tools
- Port-scanning a NetWare server
- Gleaning NetWare information without logging in
- Exploiting common vulnerabilities when logged into NetWare
- Minimizing NetWare security risks

As much as some of Novell's competitors like to say that NetWare is a thing of the past, it's still alive and kicking out there. Even though Novell is now far down the Linux path with its SUSE Linux desktop and Open Enterprise Server software, there are still tons of "classic" NetWare servers around the world. NetWare usage is certainly not without warrant — the organizations running NetWare (and other Novell products, for that matter) demand a solid directory services infrastructure and stable environment. Novell has certainly delivered in that arena.

If you do a lot of work with NetWare, now's the time to start beefing up your Linux skills! I cover Linux hacking in Chapter 11. In this chapter, I just stick with the tried and true old-school NetWare as you might know it.

NetWare administrators — arguably some of the best, most technical administrators around — often overlook or deny that NetWare is hackable. This chapter shows you how to test for the most critical NetWare exploits and outlines countermeasures to prevent the problems.

NetWare Vulnerabilities

Novell NetWare has a reputation as one of the most secure operating systems available. This is one reason that you rarely hear about NetWare servers getting hacked or having new vulnerabilities that crop up constantly. However, NetWare is not without its security issues. Various NetWare vulnerabilities can be exploited — from NDS (now called *eDirectory*) enumeration to remote password testing to spoofing NetWare packets. External attackers and malicious

insiders can exploit many of NetWare's vulnerabilities without even logging in to the server! Even a couple of Metasploit exploits can be executed in certain NetWare environments to provide remote access to unauthorized users. I cover the Metasploit penetration tool in Chapters 10 and 11.

NetWare servers are frequently the most vital servers within a network. They often perform the following functions:

- House critical files
- Store replicas of the eDirectory database for hosting, replicating, and managing such directory service objects as user IDs, printers, organizational units, application licenses, and more
- Host e-mail with Novell GroupWise
- Host Web sites and Web applications with such programs as Apache and Tomcat
- Serve as firewalls running Novell BorderManager (one of my favorite firewalls of all time!)

Choosing Tools

The following are my favorite NetWare testing tools — they offer just about everything you need to perform a solid assessment of NetWare:

- **SuperScan version 3** (version 4 is available but I like version 3 better) (`www.foundstone.com/us/resources/proddesc/superscan3.htm`) for ping sweeping and port scanning
- **LANguard** (`www.gfi.com/lannetscan`) for port scanning, OS enumeration, and vulnerability testing
- **QualysGuard** for vulnerability scanning (`www.qualys.com`)
- **Remote** (`www.securityfocus.com/data/vulnerabilities/exploits/Remote.zip`) for Remote Console password cracking

Make sure that you have the latest version of Novell's Client software from `http://download.novell.com` on your system before running these tests.

Getting Started

Although NetWare has relatively few serious security vulnerabilities, a few stand out. The hacks in this chapter are against a default installation of NetWare 5.1 from inside the firewall. However, these vulnerabilities and tests

apply to most versions of NetWare 4.*x* and newer — the ones running NDS and eDirectory. I also point out a few critical NetWare 3.*x* vulnerabilities. If you're running Novell Open Enterprise Server that's based on Linux, refer to Chapter 11.

Patches on your specific systems might have fixed some of these vulnerabilities. If you don't get the exact same results as shown in this chapter, you're probably safe.

If you have the latest Novell-supplied patches on your systems, your systems are likely secure. However, the hacks in this chapter are significant, so you should test for them to make sure that your server is safe.

Older versions of NetWare, such as 4.2 and 5.0, are being phased out of support. You'll no longer receive security updates for these versions.

Server access methods

You can access a NetWare server in the following four ways — each of which affects how you can test:

- **Not logged in:** You simply perform port scans or make NCP calls across the network without actually logging in — similar to a null session connection in the Windows world.
- **Logged in:** This connection requires you to successfully log in with a valid bindery or eDirectory user ID and password.

  Logged in is the basic method for accessing standard NetWare services.
- **Web access:** This connection might be available if you're running GroupWise WebAccess e-mail services, various NetWare management tools, or other basic Web server applications.
- **Console access:** This access method requires you either be at the server console or use a remote-connectivity product (such as NetWare's built-in rconsole or even aconsole that shipped with NetWare 3.*x* and earlier systems).

When you finish scanning your NetWare systems for open ports and general information gathering, you can test for common NetWare security vulnerabilities.

Port scanning

Start testing your NetWare systems by performing an initial port scan to check what hackers can see. You can perform these scans in two ways:

- If the server has a public IP address, scan from outside the firewall, if possible.
- If the server doesn't have a public IP address, you can scan internally on the network.

The bad guys can be inside your network, too!

The SuperScan results in Figure 12-1 show several potentially vulnerable ports open on this NetWare server, including FTP and the commonly exploited Echo and Character Generator ports. In addition, the NetWare-specific port 524 is NCP (NetWare Core Protocol). NetWare uses this protocol for its internal communications with hosts, such as clients and other servers — similar to SMB in Windows.

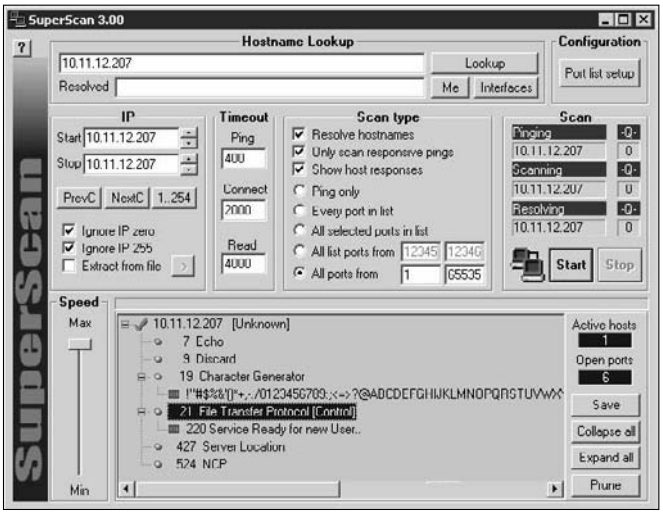


Figure 12-1: Using SuperScan to scan a default installation of NetWare 5.1.

You might also find that GroupWise is running (TCP/UDP port 1677), as well as a Web server and other Web-based remote-access ports, such as 80, 443, 2200, 8008, and 8009.

You can also perform a scan with LANguard Network Security Scanner. Using a commercial tool such as this can often provide more details about the systems you scan than a basic port scanner can. Figure 12-2 shows that LANguard can determine more information about the server, such as the NetWare version and SNMP information. This is another good use for the SNMP enumeration tool Getif (`www.wtcs.org/snmp4tpc/getif.htm`), which I describe in Chapter 8. It also tells you what's listening on the open ports without you having to look them up.

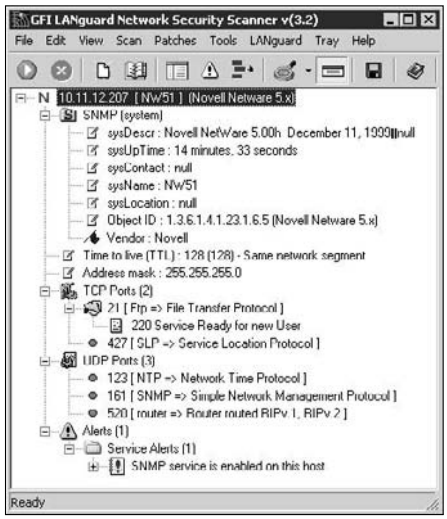


Figure 12-2: Gathering details with LANguard Network Security Scanner.

Don't overlook QualysGuard (`www.qualys.com`) as a good NetWare security testing tool. This tool tests for a handful of NetWare-specific vulnerabilities related to the NetWare Enterprise Web Server and other *abend* (a Novell term that stands for abnormal end) issues that most other tools simply don't catch.

Authentication

If attackers or malicious users can gather server details, such as server, eDirectory, and user ID information, they might be able to exploit a known vulnerability or even try to log in by using the user IDs that they discover. When they're in, all bets are off, and anything goes. They could

- Log in to your network as a regular user.
- Log in to your network as admin.
- Obtain physical access to the server console.

Testing for the worst-case scenario is wise because attackers could log in as users or administrators on your NetWare system.

rconsole

One of the most serious NetWare security vulnerabilities is the NetWare Remote Console program (referred to as *rconsole*). rconsole is an SPX protocol–based remote control program similar to telnet and Windows Terminal Services. The

program gives users full access to the NetWare console if they know the password. rconsole consists of the following:

- The remote NetWare Loadable Module (NLM) and rspx NLM files on the server
- The `rconsole.exe` client program in the `sys:\public` directory
- For rconsole to work, you must load the rspx NLM using one of these methods:
  - Enter `load rspx` at the console.
  - Place `load rspx` in your `autoexec.ncf` or `ldremote.ncf` file just below your *load remote* line.

rconsole attacks

rconsole is vulnerable because its passwords can be easily obtained. The passwords are stored in either cleartext or an easily crackable hash format on the server in the `sys:\system\autoexec.ncf` or `sys:\system\ldremote.ncf` files.

If you encrypt your rconsole passwords, cracking them is simple. The following steps show you how to set up an rconsole password so you can see just how vulnerable the rconsole password really is:

1. **Type** load remote **at the server console to load the remote NLM on the server.**
2. **Enter the password you want to use when prompted.**
3. **Type** remote encrypt **and enter your rconsole password again when prompted.**

   The server generates the encrypted password and displays the entire command you need to run on the screen, including the hashed password. It looks similar to the response in Figure 12-3.

   The server might also enter the command into the `ldremote.ncf` file, but it sometimes fails. For simplicity, just enter the load remote `-E` *`password`* command manually into your `autoexec.ncf` file. Don't write this password down and leave it where others can find it!

Now try cracking the encrypted rconsole password. For this, I use the *remote* cracking program — not to be confused with the remote NLM that's part of rconsole.

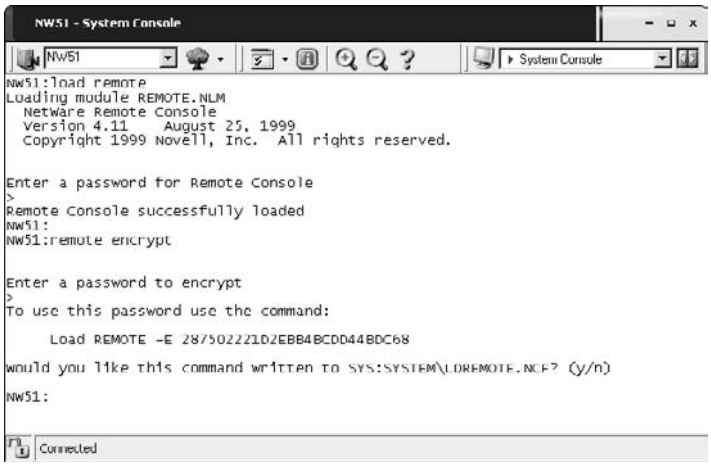


Figure 12-3: Encrypting your rconsole password.

Simply run the `remote.exe` cracking program against the rconsole password hash that's displayed on the screen (or stored in the server's `autoexe.ncf` or `ldremote.ncf` file). Enter a line similar to the following at a command prompt:

```
remote password_hash
```

The result is the rconsole password.

You can try the preceding steps against *my* password. Figure 12-3 shows the hash:

```
287502221D2EBB4BCDD44BDC68
```

Anyone using the following three items can even capture the encrypted rconsole password traveling across the wire and decrypt it:

- Network analyzer
- Rcon program (`http://packetstormsecurity.nl/Netware/penetration/rcon.zip`)
- The steps outlined in the `rconfaq.txt` file at `http://packetstormsecurity.nl/Netware/audit/rconfaq.zip`

The remote NLM stores its password in server memory. Anyone with console access can go into the NetWare debugger by pressing Shift+Alt+Shift+Esc (yes, you use both Shift keys) on the server keyboard and can view it in cleartext.

Countermeasures against rconsole attacks

The following can prevent attacks against NetWare servers:

- **Don't use rconsole.** At least, don't use it on critical NetWare servers. (Aren't all servers critical, though?)
- **If you must use rconsole, secure it with one of the following steps for your version of NetWare:**
  - For NetWare 4.*x* or earlier, lock your server by using the monitor NLM.
  - With NetWare 5 and newer, load the scrsaver NLM. It displays the fancy text-based NetWare snake and requires a valid NetWare account to unlock.
- **Consider using one of these remote NetWare management programs instead of rconsole:**
  - Rconj is a Java-based version of rconsole that works over TCP. It comes with NetWare 5.*x* and later, but has limited functionality.

    Be sure to patch Rconj if you run it on NetWare 6. Rconj has a known authentication vulnerability when running on NetWare 6 that allows a hacker to gain access without a password.

Server-console access

Physical access to the server console is a hacker's pot of gold. After hackers obtain this access, they can do practically anything they want to with the server. They can access the NetWare debugger to retrieve passwords and potentially other confidential information stored in memory — not to mention crash the server.

The following countermeasures help ensure that NetWare console access is confined to those who are authorized:

- **Physical security (such as the usage of server locks) is a must.** Chapter 6 explains how to test and subsequently secure server rooms and data centers.
- **Lock the server screen.** You can keep the server console secure by either selecting the Lock Server Console option in the monitor NLM or loading the scrsaver NLM.

Intruder detection

Intruder detection is one of the most critical security features built in to NetWare. It locks a user account for a specific period of time after a certain number of failed login attempts.

Make sure that intruder detection is enabled on your system. It's *disabled* by default.

Testing for intruders

Default settings for intruder detection — after it's enabled — in NetWare 5.1 are shown in Figure 12-4. Chapter 7 details intruder detection and password lockout.

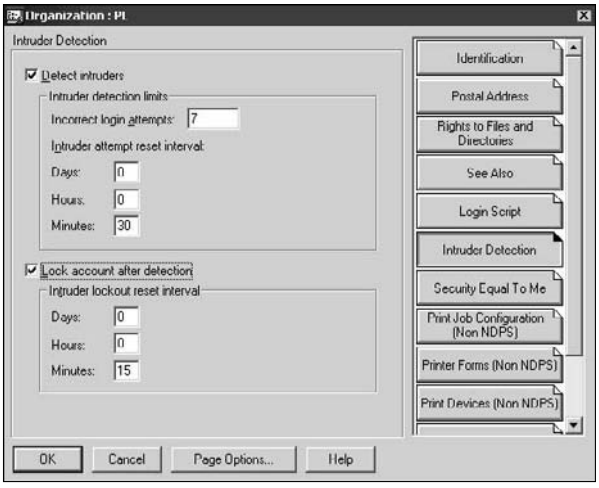


Figure 12-4: Intruder-detection settings in NetWare 5.1.

Try logging in with invalid passwords for several test users — preferably, users from different organizational units (OUs) within eDirectory — to see whether intruder detection is working. Make sure that you type *bad* passwords; blank ones don't seem to work well for this test. Here's how you know whether intruder detection is working:

- If intruder detection is on, you should get a response similar to Figure 12-5.
- If intruder detection is off, you're prompted repeatedly for a password.

Figure 12-5: A Novell Client32 message.

Malicious attackers use this process to determine whether intrusion detection is enabled on your NetWare server.

Countermeasures against intruders

You can implement the following countermeasures to ensure that unauthorized logins are minimized and intruder detection is not abused:

- **Enable intruder detection as high in the directory tree as possible —** preferably, at the uppermost organization level.

  This is one of the best hacking countermeasures you can implement in a NetWare environment.
- **Look for evidence that the console NLM was unloaded by searching for entries in the `sys:\etc\console.log` file.**
- **Consider logging all events to a remote syslog server to help prevent a hacker from tampering with evidence. A good resource for this is `www.loganalysis.org`.**

Testing for rogue NLMs

If a hacker gains console access to your server, a legitimate yet potentially dangerous NLM can be loaded, which can do bad things to the system.

The following tests look for rogue NLMs running on your server.

Modules command

You can use the modules command at the server console prompt to view loaded modules. As shown in Figure 12-6, you simply enter the command **modules** at the server-console screen, and the server displays a listing of NLMs that are loaded, organized in order of loading.

Look for these NLMs in the modules output. If neither you nor another administrator has loaded the following NLMs, you have a problem:

- **Setpwd password reset tool:** This third-party NLM can reset *any* user's password on the server — including admin! Find it at `ftp://ftp.cerias.purdue.edu/pub/tools/novell/setpwd.zip`.

- **dsrepair:** This built-in NLM can corrupt or destroy eDirectory. It's actually intended to repair and maintain the eDirectory database.
- **netbasic:** This built-in NLM can copy eDirectory files from the hidden `sys:\_netware` directory. It accesses a DOS-like prompt on the server.

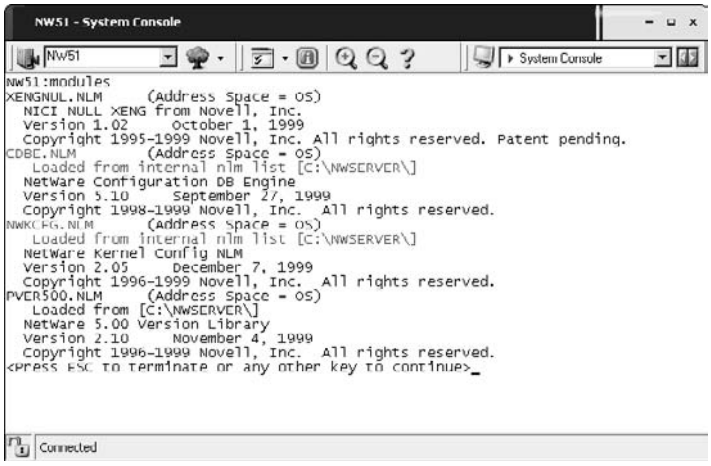


Figure 12-6: Viewing loaded applications on a NetWare server.

Check whether the nwconfig NLM is loaded. This built-in NLM is often used for day-to-day server maintenance, such as installing patches and editing system files. However, a hacker can load it and back up or restore the eDirectory database so that its files can be copied for malicious purposes. You can look to see whether the NLM is loaded by

- Looking at the modules output
- Pressing Ctrl+Esc to view all loaded applications
- Pressing Alt+Esc to toggle through all loaded applications

Many NLMs can load on a NetWare server — especially in the more recent versions. If you have a question about what an NLM does or want to see whether it's valid, you can search on the filename at `www.google.com` or at `http://support.novell.com` to get more information.

A port scan of the server from another computer can find rogue applications as well.

Tcpcon

The tcpcon NLM shows ports that are listening and connected. Follow these steps to use it:

1. **Enter** load tcpcon **at the server prompt.**
2. **Choose Protocol Information from the main menu.**
3. **Select TCP and then TCP Connections to view the open TCP ports.**
4. **Select UDP and then UDP Listeners to view the open UDP ports.**

   Figure 12-7 shows the TCP ports that are open and listening on this server, including chargen, FTP, and NCP (port 524).

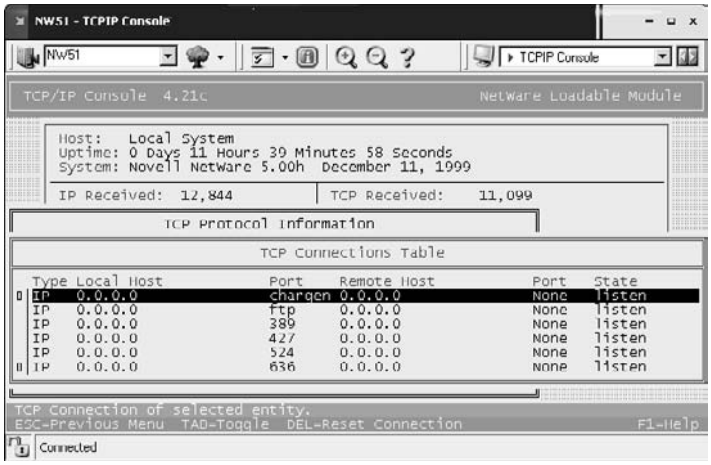


Figure 12-7: Using tcpcon to show open TCP ports on a NetWare server.

If something doesn't look right, it might not be, so investigate the port number further. My favorite port number reference is at `www.iana.org/assignments/port-numbers`, but a simple Google search is usually productive.

Admin utilities

If hackers can successfully log in to a NetWare server or eDirectory, they can use, in malicious ways, some of the great — and free — NetWare admin utilities from JRB Software (`www.jrbsoftware.com`). For example, hackers can

- Run the downsrvr program to reboot a NetWare server — most likely at the worst possible time.
- Use the serv_cmd program to disable logins, remotely load NLMs, and add bindery contexts to the system.

Countermeasures against rogue NLM attacks

The following countermeasures can minimize the chances that malicious NLMs can run on your servers.

Documentation

The best way to keep track of loaded NLMs is to document, document, and document your server. Knowing what's supposed to be loaded on your server at all times is critical.

- **For each loaded NLM, you need to know its name, version, and date.**
- **Save and print recent versions of your `startup.ncf` and `autoexec.ncf` files.**
- **Document — at least, at a high level — your eDirectory structure.** You can either
  - Take a screen capture of eDirectory as it looks in NetWare Administrator or ConsoleOne.
  - Run `cx /t /a /r` and save the output of the program to a text file by entering the following at a command prompt:

```
cx /t /a /r > filename.txt
```

Update your documentation after any system changes are made or any new patches are applied.

Unauthorized logins

To prevent rogue NLMs or remote applications from being loaded or run from a workstation, apply these security measures to your NetWare systems:

- **Make strong passwords on *every* NetWare account.** I outline minimum password requirements in Chapter 7.
- **Secure the server console.**
- **Enable intruder detection.**
- **Neutralize dangerous NLMs, such as netbasic.** You can either rename them or remove them.

  If you remove dangerous NLMs, make a backup of the files first. You might need them in the future.

Cleartext packets

Most internal LAN traffic — regardless of the operating system in use — travels across the wire in cleartext by default. The cleartext can be captured and used against you.

Packet capture

Cleartext packets can be captured with one of the following:

- A network analyzer
- Components of the Pandora NetWare hacking suite (`www.nmrc.org/project/pandora`)

Pandora can spoof NCP packets, which can give attackers admin equivalency on the network after they log in via standard user accounts that they previously compromised. Hackers can log in as normal users with a weak or blank password and then use Pandora to manipulate NetWare traffic and get admin rights on the network.

Countermeasures against packet capture

You can easily set up *NCP packet signing* within a NetWare environment. This tool encrypts and provides proof that a packet actually originated from the sending host. NCP packet signing has four levels, but the level for the utmost security is level 3, which requires packet signatures.

This tool can slow network traffic and place a larger processing burden on your server. Level-3 packet signing can decrease network performance on busy NetWare servers — sometimes by more than 50 percent.

The following steps explain how to enable level-3 packet signing:

- Enable level-3 packet signing on the server and at the top of the `autoexec.ncf` file with the following command:

```
set ncp packet signature option=3
```

- Enable level-3 packet signing on NetWare clients with these steps:
  1. *Right-click the red Novell icon in the Windows system tray.*
  2. *Choose Novell Client Properties and Advanced Settings.*
  3. *Set the Signature Level to 3 (Required).*

In NetWare 3.*x* and earlier, passwords are sent in cleartext across the network. For these versions, you can enter the following command on your server and in the `autoexec.ncf` file to help prevent passwords from being captured with a network analyzer:

```
set allow unencrypted passwords=off
```

Solid Practices for Minimizing NetWare Security Risks

Although you can't *completely* defend NetWare servers against attacks, you can come pretty close, which is an improvement over other "leading" operating systems. These NetWare hacking countermeasures can help improve security on your NetWare server beyond what I've already recommended.

Rename admin

Rename the admin account. Figure 12-8 shows how this can be done in the Novell ConsoleOne utility.

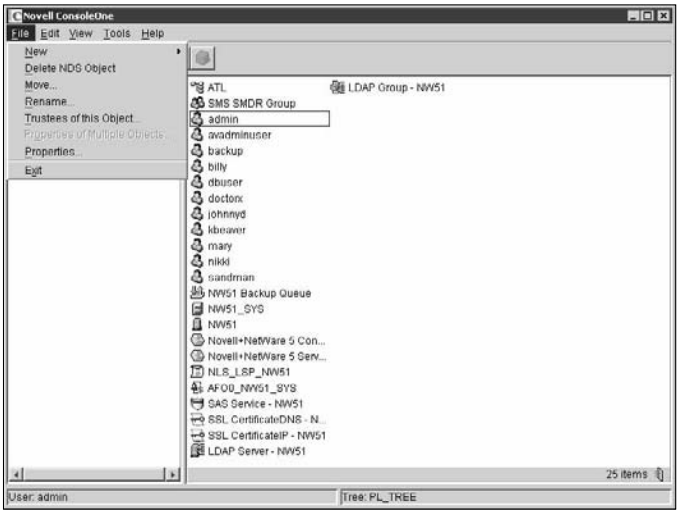


Figure 12-8: Renaming the NetWare admin account with ConsoleOne.

Be careful when renaming the admin account, though. Other applications, such as the server backup software, might depend on the admin ID.

If you rename admin, be sure to edit any backup jobs or startup scripts that depend on the admin account name. It's best to not use the admin account for backup and other administrative tasks anyway, so this might be a good time to make a change by creating an admin equivalent for each application that depends on an admin ID. Creating these equivalents can help make your system more secure by reducing the number of places that the admin account is exposed and vulnerable to cracking on the network.

Disable eDirectory browsing

Disabling Public's right to browse the directory tree in either NetWare Administrator for NetWare 4.*x* or Novell ConsoleOne for NetWare 5.*x* and later is a good way to ward off attacks. This right is enabled by default to allow users to browse the eDirectory tree easily.

Disabling the Public Browse right or any other eDirectory or file rights can cause problems, such as locking users (including you) out of the network, disabling login scripts, and disabling printing. The potential risk depends on how you configure eDirectory. If you remove Public's Browse right, you can usually grant specific object rights lower in the tree, where they're needed to keep everything working. Make sure that you test these types of critical changes before applying them to your production environment.

NetWare Administrator

Follow these steps to disable the Public Browse right to eDirectory with NetWare Administrator (`sys:\public\win32\nwadmn32.exe`):

1. **Right-click the Root object in your directory tree.**
2. **Choose Trustees of This Object.**
3. **Select the [Public] trustee, as shown in Figure 12-9.**
4. **In the Object Right section, deselect the Browse check box.**
5. **Click OK.**

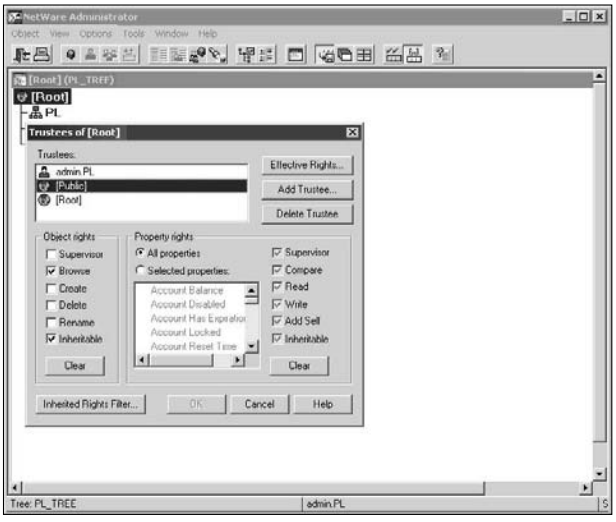


Figure 12-9: The default Browse right for [Public], shown in NetWare Administrator.

Novell ConsoleOne

Follow these steps to disable the Public Browse right to eDirectory with Novell ConsoleOne (`sys:\public\mgmt\ConsoleOne\1.2\bin\ConsoleOne.exe`):

1. **Right-click your tree object.**
2. **Choose Trustees of This Object.**
3. **Select the [Public] trustee and then click Assigned Rights.**
4. **In the Rights section, deselect the Browse check box, as shown in Figure 12-10.**
5. **Click OK twice.**

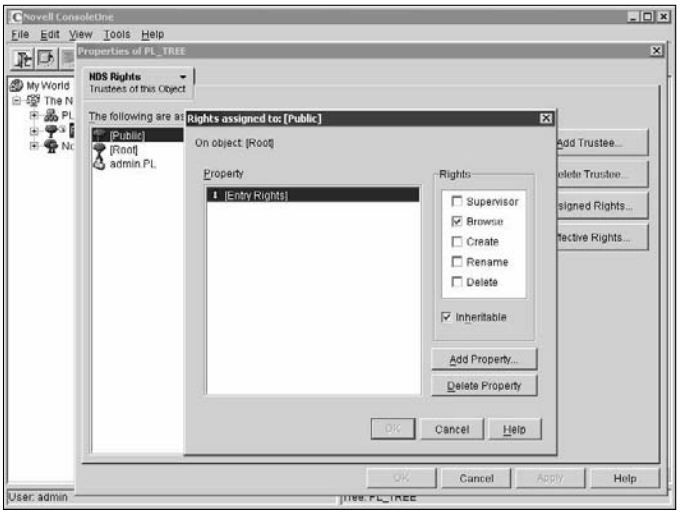


Figure 12-10: The default Browse right for [Public], shown in ConsoleOne.

Remove bindery contexts

Remove any bindery contexts loaded on your server. Bindery contexts are in place in NetWare 4.*x* and later to provide backward compatibility with older clients that need to access the servers as though they're NetWare 3.*x* or earlier servers. This is typically present (and necessary) for older applications or NetWare clients (such as netx and VLMs) that make bindery calls instead of eDirectory calls.

Removing bindery contexts can help prevent hacker attacks against bindery weaknesses. To disable the bindery context on your server, simply comment out the *set Bindery Context* line in your server's `autoexec.ncf` file using a # sign.

If you remove your bindery contexts, make sure that no clients or applications depend on NetWare bindery emulation.

Audit the system

Turn on system auditing by running `auditcon` at a command prompt. This program can help you track down a future intruder by auditing files, volumes, and even the directory tree. It's just a good security practice, as well.

TCP/IP parameters

In NetWare 5.*x* and above, based on your specific version, you can prevent several types of DoS attacks by entering the following TCP/IP parameters at the server console:

```
set discard oversized ping packets=on
set discard oversized UDP packets=on
set filter subnet broadcast packets=on
set filter packets with IP header options=on
set ipx netbios replication option=0
set tcp defend land attacks=on
set tcp defend syn attacks=on
```

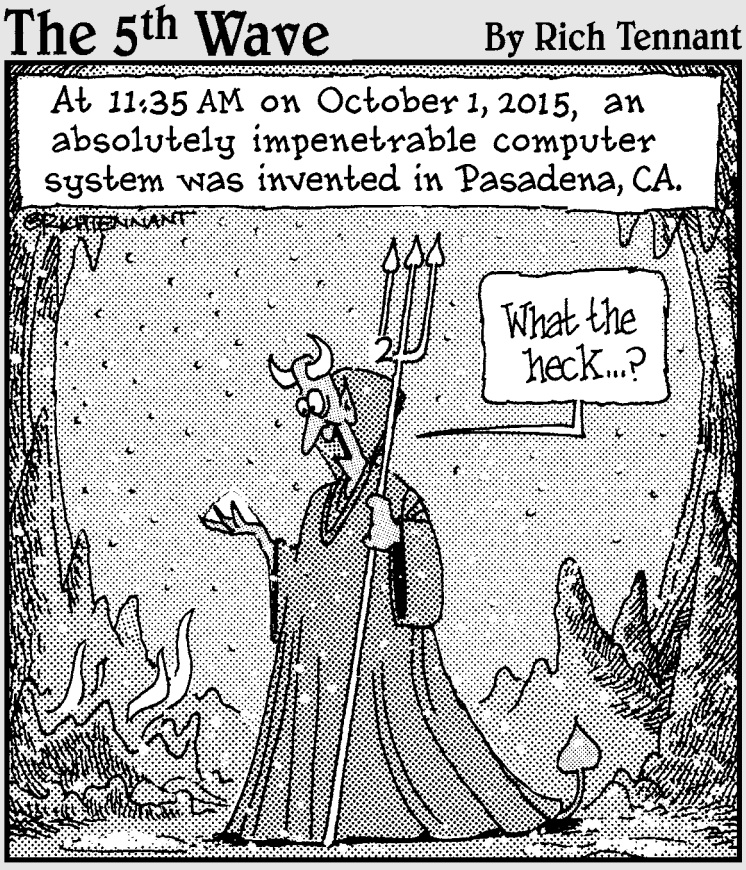
You can enter the preceding commands into the server's `autoexec.ncf` file so that they load each time the server starts.

Patch

Patch, patch, and patch again! Novell lists the latest patches for the NetWare versions it supports on its Web site:

```
http://download.novell.com
```

Part V
Hacking Applications



In this part . . .

Well, this book has covered everything from non-technical hacks to network hacks to operating system hacks. What I haven't yet covered are the applications that run on top of all of this and the storage systems that keep everything intact.

The first chapter in this part covers various messaging hacks and countermeasures affecting e-mail, instant messaging, and Voice over IP (VoIP) systems. Next, this part looks at Web application exploits, along with some countermeasures to secure them from the elements. It then goes deeper into application hacking using Google and its searching capabilities. Finally, this part covers attacks against storage systems. It covers both unstructured data, otherwise known as *network files,* and structured data found in various database systems.

Chapter 13

Communication and Messaging Systems

In This Chapter

- Attacking e-mail systems
- Assailing instant messaging
- Assaulting Voice over IP applications

Messaging systems — you know, those e-mail, instant messaging (IM), and Voice over IP (VoIP) applications that we all depend on — often create vulnerabilities that people overlook. Why? Well, from my experience, messaging software — both at the server and client level — is vulnerable because network administrators often believe that firewalls and antivirus software are all that's needed to keep trouble away, or they simply forget about securing these systems altogether.

In this chapter, I show you how to test for common e-mail, IM, and VoIP issues. I also outline key countermeasures to help prevent these hacks against your systems.

Messaging System Vulnerabilities

Practically all messaging applications are hacking targets on your network. E-mail systems are some of the most targeted. Given the proliferation and

business value of IM and other P2P applications, attacks against networks launched via IM channels are about as common as e-mail attacks. Wondering about VoIP? Well, it's downright scary what people with ill intent can do with it.

Tons of vulnerabilities are inherent in messaging systems. This is because most messaging protocols weren't designed with security in mind — especially those developed several decades ago when security wasn't nearly the issue it is today. The funny thing is that even modern-day messaging protocols — or at least the implementation of the protocols — used in IM and VoIP are *still* susceptible to serious security problems. Furthermore, convenience and usability often outweigh the need for security.

Many attacks against messaging systems are just minor nuisances; others can inflict serious harm on your information and your organization's reputation. Malicious attacks against messaging systems include

- Transmitting malware
- Crashing servers
- Obtaining remote control of workstations
- Capturing sensitive information while it travels across the network
- Perusing e-mails stored on servers and workstations
- Perusing IM log files on workstation hard drives
- Gathering messaging trend information via log files or a network analyzer that can tip off the attacker about conversations between people and organizations
- Capturing and replaying phone conversations
- Gathering internal network configuration information, such as hostnames and IP addresses

These attacks can lead to such problems as unauthorized — and potentially illegal — disclosure of sensitive information, and loss of information altogether.

A case study in e-mail hacking with Thomas Akin

In this case study, Thomas Akin, a well-known expert in e-mail systems and forensics, shared with me an experience in e-mail hacking.

The Situation

Mr. Akin was involved in a case in which a client's e-mail system was blacklisted for sending hundreds of thousands of spam e-mails. The client spent two weeks reconfiguring the e-mail server in an attempt to stop the spam e-mails from going through the system. The client looked at every technical possibility — including making sure that the server was not an open SMTP relay — but nothing worked. Over 100,000 spam e-mails a day were being sent through the company. After losing several customers because the company couldn't send them any e-mails, the company called Mr. Akin to see whether he could help.

Mr. Akin first checked to see whether the e-mail system was acting as an open relay, but it was not. Because the e-mail system wasn't misconfigured, there shouldn't have been any reason for blacklisting the client. Then Mr. Akin reviewed the spam e-mail headers, expecting to see a standard spoofed e-mail. Instead, after reviewing the headers, he saw that they *were* coming from the company's e-mail system. Not only that, but they were also originating from a reserved IP address — an address that isn't even allowed on the Internet.

Momentarily stumped, Mr. Akin looked at the text of the e-mail messages themselves. "One time only!" "Buy me now!" "Best deal ever!" This is the standard spam nonsense, except that these e-mails were signed by Laura and John (names disguised to protect the guilty). Not only that, Laura and John listed their phone numbers so potential customers could contact them easily. How nice of them!

The Outcome

A quick search online turned up a phone number match to a Laura and John living in East Bumble, USA. Bingo! It turned out that John was a former employee and that his dial-up account had not been disabled when he was fired from the company. A quick glance at the log files showed that the "john" account had used the company's dial-up access during the exact times the spam e-mails were sent. The company immediately disabled the account, and the spam e-mails stopped.

Even though the spamming was stopped, the company was desperate to know how the e-mails were being sent through its system. The dial-up account should have allowed only limited access through a menu system — not full access to the organization's network. After some research, Mr. Akin determined that John had bypassed the dial-up's menu system and was using a program called slirp to turn his internal dial-up connection into a full Internet connection. Because John was dialing into the company's modem bank, the e-mail system saw him as an internal user, letting him send e-mail to anyone and anywhere he wanted. The company quickly reviewed all dial-up accounts and found that over two dozen accounts were still active and being used by former employees!

Thomas Akin was the founding director of the Southeast Cybercrime Institute at Kennesaw State University and is a member of the X-Force Emergency Response Team at Internet Security Systems. Mr. Akin is a CISSP, holds several networking certifications, and is a member of Mensa.

E-Mail Attacks

The following attacks exploit the most common e-mail security vulnerabilities I've seen. The good news is that you can eliminate or minimize most of them to the point where your information is not at risk. You might not want to carry out all these attacks against your e-mail system — especially during peak traffic times — so be careful!

Some of these attacks require the basic hacking methodologies: gathering public information, scanning and enumerating your systems, and finding and exploiting the vulnerabilities. Others can be carried out by sending e-mails or capturing network traffic.

E-mail bombs

E-mail bombs can crash a server and provide unauthorized administrator access. They attack by creating denial of service (DoS) conditions against your e-mail software and even your network and Internet connection by taking up a large amount of bandwidth and, sometimes, requiring large amounts of storage space.

Attachments

An attacker can create an attachment-overload attack by sending hundreds or thousands of e-mails with very large attachments to one or more recipients on your network.

Attacks using e-mail attachments

Attachment attacks have a couple of goals:

- The whole e-mail server might be targeted for a complete interruption of service with these failures:
  - *Storage overload:* Multiple large messages can quickly fill the total storage capacity of an e-mail server. If the messages aren't automatically deleted by the server or manually deleted by individual user accounts, the server will be unable to receive new messages.

    This can create a serious DoS problem for your e-mail system, either crashing it or requiring you to take your system offline to clean up the junk that has accumulated. A 100MB file attachment sent ten times to 100 users can take 100GB of storage space. Yikes!

  - *Bandwidth blocking:* An attacker can crash your e-mail service or bring it to a crawl by filling the incoming Internet connection with junk. Even if your system automatically identifies and discards obvious attachment attacks, the bogus messages eat resources and delay processing of valid messages.

- An attack on a single e-mail address can have serious consequences if the address is for an important user or group.

Countermeasures against e-mail attachment attacks

These countermeasures can help prevent attachment-overload attacks:

- **Limit the size of either e-mails or e-mail attachments.** Check for this option in your e-mail server's configuration settings (such as those provided in Novell GroupWise and Microsoft Exchange), your e-mail content filtering system, and even at the e-mail client level.
- **Limit each user's space on the server.** This denies large attachments from being written to disk. Limit message sizes for inbound and even outbound messages should you want to prevent a user from launching this attack from inside your network. I find 500MB is a good limit, but it all depends on your network size, storage availability, business culture, and so on, so think through this one carefully before putting anything in place.

  Consider using FTP or HTTP instead of e-mail for large file transfers, and encourage internal users to use departmental shares or public folders. By doing so, you can store one copy of the file on a server and have the recipient download the file on his or her own workstation.

Contrary to popular logic and use, the e-mail system should *not* be an information repository but that's how e-mail is evolving. An e-mail server used for this purpose can create unnecessary legal and regulatory risks and can turn into an absolute nightmare if your business receives an e-discovery request related to a lawsuit.

Connections

A hacker can send a huge number of e-mails simultaneously to addresses on your network. These connection attacks can cause the server to give up on servicing any inbound or outbound TCP requests. This can lead to a complete server lockup or a crash, often resulting in a condition in which the attacker is allowed administrator or root access to the system.

Attacks using floods of e-mails

This attack is often carried out in spam attacks and other denial of service attempts.

Countermeasures against connection attacks

Many e-mail servers allow you to limit the number of resources used for inbound connections, as shown in the Number of SMTP Receive Threads option for Novell GroupWise in Figure 13-1. This setting is called different things for different e-mail servers and e-mail firewalls, so check your documentation. Completely stopping an unlimited number of inbound requests is impossible. However, you can minimize the impact of the attack. This setting limits the amount of server processor time, which can help during a DoS attack.

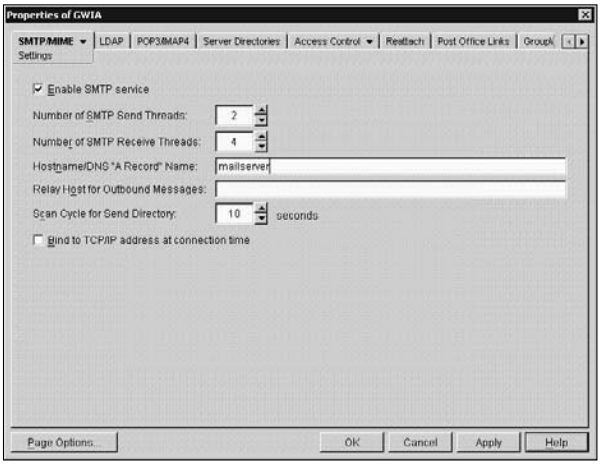


Figure 13-1: Limiting the number of resources that handle inbound messages.

Even in large companies, there's no reason that thousands of inbound e-mail deliveries should be necessary within a short time period.

Some e-mail servers, especially UNIX-based servers, can be programmed to deliver e-mails to a daemon or service for automated functions, such as *create this order on the fly when a message from this person is received*. If DoS protection isn't built in to the system, a hacker can crash both the server and the application that receives these messages and potentially create e-commerce liabilities and losses. This can happen more easily on e-commerce Web sites when CAPTCHA (short for Completely Automated Public Turing Test to Tell Computers and Humans Apart) is not used on forms. I cover Web application security in Chapter 14.

Prevent e-mail attacks as far out on your network perimeter as you can. The more traffic or malicious behavior you keep off your e-mail servers and clients, the better.

Automated e-mail security controls

You can implement the following countermeasures as an additional layer of security for your e-mail systems:

- **Tarpitting:** *Tarpitting* detects inbound messages destined for unknown users. If your e-mail server supports tarpitting, it can help prevent spam or DoS attacks against your server. If a predefined threshold is exceeded — say, more than ten messages — the tarpitting function effectively blocks traffic from the sending IP address for a period of time.

- **E-mail firewalls:** E-mail firewalls and content-filtering applications, such as CipherTrust IronMail (`www.mcafee.com/us/enterprise/products/email_and_web_security/email/email_gateway.html`) and Messaging Architect's M+Guardian (`www.messagingarchitects.com/products/m-guardian-email-security.html`) can prevent various e-mail attacks. These tools protect practically every aspect of an e-mail system. Given today's e-mail threats, one of these is a must for any serious network manager.
- **Perimeter protection:** Although not e-mail-specific, many firewall, IDS, and IPS systems can detect various e-mail attacks and shut off the attacker in real time. This can come in handy during an attack at an inconvenient time.
- **CAPTCHA:** Using CAPTCHA on Web-based e-mail forms can help prevent automated attacks and lessen your chances of e-mail flooding and denial of service. These benefits come in handy when scanning your Web sites and applications, as I discuss in Chapter 14.

Banners

When hacking an e-mail server, a hacker's first order of business is performing a basic banner grab to see whether he can discover what e-mail server software is running. This is one of the most critical tests to find out what the world knows about your SMTP, POP3, and IMAP servers.

Gathering information

Figure 13-2 shows the banner displayed on an e-mail server when a basic telnet connection is made on port 25 (SMTP). To do this, at a command prompt, simply enter **telnet *ip or_hostname_of_your_server* 25**. This opens a telnet session on TCP port 25.

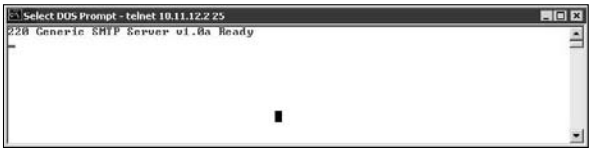


Figure 13-2: An SMTP banner showing server-version information.

The e-mail software type and server version is often very obvious and gives hackers some ideas about possible attacks, especially if they search a vulnerability database for known vulnerabilities of that software version. Figure 13-3 shows the same e-mail server with its SMTP banner changed from the default

(okay, the previous one was, too) to disguise such information as the e-mail server's version number.

Figure 13-3: An SMTP banner that disguises the version information.

You can gather information on POP3 and IMAP e-mail services by telnetting to port 110 (POP3) or port 143 (IMAP).

If you change your default SMTP banner, don't think that no one can figure out the version. One Linux-based tool called smtpscan (`www.freshports.org/security/smtpscan`) determines e-mail server version information based on how the server responds to malformed SMTP requests. Figure 13-4 shows the results from smtpscan against the same server shown in Figure 13-3. The smtpscan tool detected the product and version number of the e-mail server.

Figure 13-4: smtpscan gathers version info even when the SMTP banner is disguised.

Common vulnerability scanners, such as QualysGuard and LANGuard, can be used to determine banner information. Using such tools also provides another benefit: finding security holes in the specific e-mail server software you're running, such as the heap overflows in Microsoft Exchange (`http://cve.mitre.org/cgi-bin/cvename.cgi?name=2006-0027`) and Novell NetMail (`http://cve.mitre.org/cgi-bin/cvename.cgi?name=2006-6424`) — both of which are exploitable by using Metasploit. I outline how to use Metasploit to exploit such vulnerabilities in detail in Part IV.

Countermeasures against banner attacks

There isn't a 100 percent secure way of disguising banner information. I suggest these banner security tips for your SMTP, POP3, and IMAP servers:

- **Change your default banners to cover up the information.**
- **Make sure that you're always running the latest software patches.**
- **Harden your server as much as possible** by using well-known best practices from such resources as the Center for Internet Security (`www.cisecurity.org`), NIST (`http://csrc.nist.gov`), and *Network Security For Dummies,* by Chey Cobb.

SMTP attacks

Some attacks exploit weaknesses in the Simple Mail Transfer Protocol (SMTP). This e-mail communication protocol — which is a quarter-century old — was designed for functionality, not security.

Account enumeration

A clever way that attackers can verify whether e-mail accounts exist on a server is simply to telnet to the server on port 25 and run the `VRFY` command. The `VRFY` — short for verify — command makes a server check whether a specific user ID exists. Spammers often automate this method to perform a *directory harvest attack* (DHA), which is a way of gleaning valid e-mail addresses from a server or domain so hackers know whom to send spam, phishing, or malware-infected messages to.

Attacks using account enumeration

Figure 13-5 shows how easy it is to verify an e-mail address on a server with the `VRFY` command enabled. Scripting this attack can test thousands of e-mail address combinations.

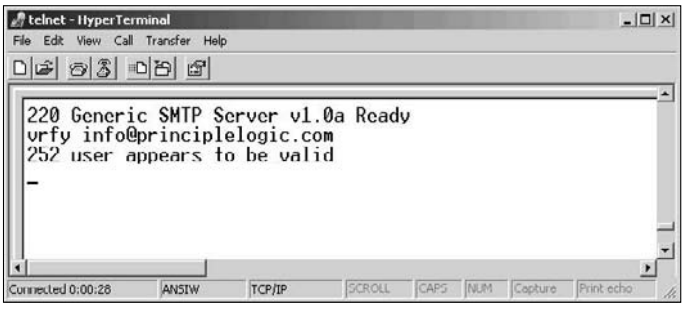


Figure 13-5: Using VRFY to verify that an e-mail address exists.

The SMTP command `EXPN` — short for expand — might allow attackers to verify what mailing lists exist on a server. You can simply telnet to your e-mail server on port 25 and try `EXPN` on your system if you know of any mailing lists that might exist. Figure 13-6 shows how the result might look. Scripting this attack and testing thousands of mailing-list combinations is simple.

Figure 13-6: Using EXPN to verify that a mailing list exists.

You might get bogus information from your server when performing these two tests. Some SMTP servers (such as Microsoft Exchange) don't support the `VRFY` and `EXPN` commands, and some e-mail firewalls simply ignore them or return false information.

Another way to somewhat automate the process is to use the EmailVerify program in TamoSoft's Essential NetTools. As shown in Figure 13-7, you simply enter an e-mail address, click Start, and EmailVerify connects to the server and pretends to send an e-mail.

Yet another way to capture valid e-mail addresses is to use TheHarvester (known as Goog Mail Enum) that's part of the BackTrack toolset to glean addresses via Google and other search engines. As I outline in Chapter 8, you can download BackTrack from `www.remote-exploit.org/backtrack.html` to burn the ISO image to CD or boot the image directly through VMWare or VirtualBox. In the BackTrack GUI, simply choose Backtrack➪Information Gathering➪SMTP➪Goog Mail Enum and enter **./goog-mail.py –d <your_domain_name> -l 500 –b google,** as shown in Figure 13-8.

If you're running BackTrack version 3 and are having trouble gleaning e-mails from Google, you need to edit the `goog-mail.py` file (via Kedit or similar) and change the following line in the Google section from this:

```
data=re.sub('<b>','',data)
```

To this:

```
data=re.sub('<em>','',data)
```

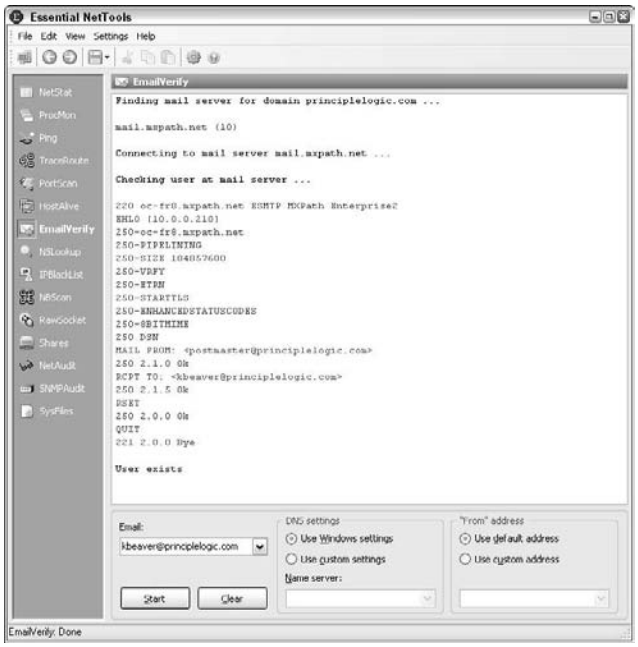


Figure 13-7: Using EmailVerify to verify an e-mail address.

Countermeasures against account enumeration

If you're running Exchange, account enumeration won't be an issue. If you're not running Exchange, the best solution for preventing this type of e-mail account enumeration depends on whether you need to enable the `VRFY` and `EXPN` commands:

- Disable `VRFY` and `EXPN` unless you need your remote systems to gather user and mailing-list information from your server.
- If you need `VRFY` and `EXPN` functionality, check your e-mail server or e-mail firewall documentation for the ability to limit these commands to specific hosts on your network or the Internet.

Figure 13-8: Goog Mail Enum for gleaning e-mail addresses via Google.

Finally, work with your marketing team and Web developers to ensure that company e-mail addresses are not posted on the Web. Educate your users about not doing this, too.

Relay

SMTP relay lets users send e-mails through external servers. Open e-mail relays aren't the problem they used to be, but you still need to check for them. Spammers and hackers can use an e-mail server to send spam or malware through e-mail under the guise of the unsuspecting open-relay owner.

Be sure to test for open relay from outside your network. If you test from inside, you might get a false positive because outbound e-mail relaying might be configured and necessary for your internal e-mail clients to send messages to the outside world. However, if a client system is compromised, this could be just what the bad guys need to launch a spamming attack.

Automatic testing

Here are a couple of easy ways to test your server for SMTP relay:

- **Free online tools.** One of my favorite online tools is located at `www.abuse.net/relay.html`.
- **Windows-based tools,** such as NetScanTools Pro (`www.netscantools.com`). You can run an SMTP Relay check on your e-mail server with NetScanTools Pro, as shown in Figure 13-9.

  Although some SMTP servers accept inbound relay connections and make it look like relaying works, this isn't always the case because the initial connection might be allowed, but the filtering actually takes place behind the scenes. Check whether the e-mail actually made it through by checking the account you sent the test relay message to.

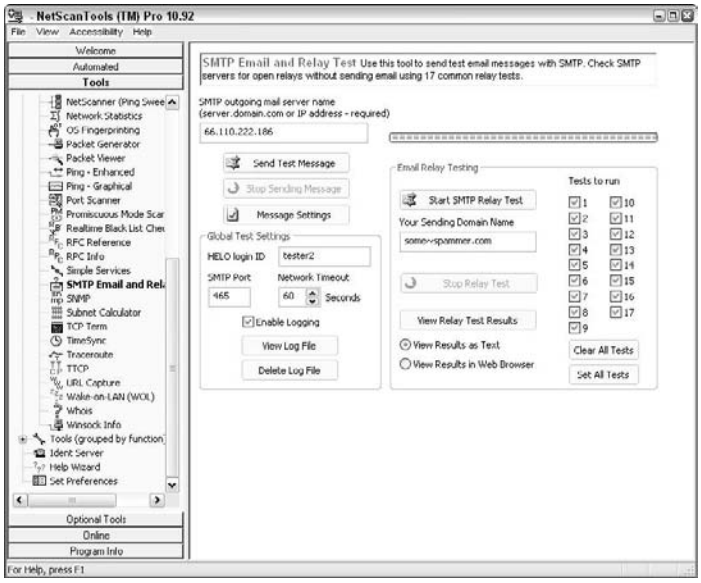


Figure 13-9: SMTP relay check tool in NetScanTools Pro for Windows.

Manual testing

You can manually test your server for SMTP relay by telnetting to the e-mail server on port 25. Follow these steps:

1. **Telnet to your server on port 25.**

   You can do this two ways:

   - Use your favorite graphical telnet application, such as HyperTerminal (which comes with Windows) or SecureCRT (`www.vandyke.com`).
   - Enter the following command at a Windows or UNIX command prompt:

```
telnet mailserver_address 25
```

You should see the SMTP welcome banner when the connection is made.

2. **Enter a command to tell the server, "Hi, I'm connecting from this domain."**

   After each command in these steps, you should receive a different-numbered message, like `999 OK`. You can ignore these messages.

3. **Enter a command to tell the server your e-mail address, for example:**

```
mail from:yourname@yourdomain.com
```

4. **Enter a command to tell the server who to send the e-mail to, for example:**

```
rcpt to:yourname@yourdomain.com
```

5. **Enter a command to tell the server that the message body is to follow, for example:**

```
data
```

6. **Enter the following text as the body of the message:**

```
A relay test!
```

7. **End the command with a period on a line by itself.**

   You can enter **?** or **help** at the first telnet prompt to see a list of all the supported commands and, depending on the server, get help on the use of the commands.

   The final period marks the end of the message. After you enter this final period, your message will be sent if relaying is allowed.

8. **Check for relaying on your server:**

   - Look for a message similar to *Relay not allowed* coming back from the server.

     If you get a message similar to this, SMTP relaying is either not allowed on your server or is being filtered because many servers block messages that appear to originate from the outside yet come from the inside.

     You might get this message after you enter the `rcpt to:` command.

   - If you don't receive a message from your server, check your Inbox for the relayed e-mail.

     If you receive the test e-mail you sent, SMTP relaying is enabled on your server and probably needs to be disabled. The last thing you want is to let spammers or other attackers make it look like you're sending tons of spam, or worse, to be blacklisted by one or more of the blacklist providers, which disrupts e-mail sending and receiving altogether.

Countermeasures against SMTP relay attacks

You can implement the following countermeasures on your e-mail server to disable or at least control SMTP relaying:

- **Disable SMTP relay on your e-mail server.** If you don't know whether you need SMTP relay, you probably don't. You can enable SMTP relay for specific hosts on the server or within your firewall configuration.

  `www.mail-abuse.com/an_sec3rdparty.html` provides information on disabling SMTP relay on e-mail servers.

- **Enforce authentication if your e-mail server allows it.** You might be able to require password authentication on an e-mail address that matches the e-mail server's domain. Check your e-mail server and client documentation for details on setting up this type of authentication.

E-mail header disclosures

If your e-mail client and server are configured with typical defaults, a malicious attacker might find critical pieces of information:

- Internal IP address of your e-mail client machine (which can lead to the enumeration of your internal network)
- Software versions of your client and e-mail server along with their vulnerabilities
- Hostnames

Testing

Figure 13-10 shows the header information revealed in a test e-mail I sent to my free Web account. As you can see, it shows off quite a bit of information about my e-mail system:

- The third Received line discloses my system's hostname, IP address, server name, and e-mail client software version.
- The X-Mailer line displays the Microsoft Outlook version I used to send this message.

Countermeasures against header disclosures

The best countermeasure to prevent information disclosures in e-mail headers is to configure your e-mail server or e-mail firewall to rewrite your headers, by either changing the information shown or removing it. Check your e-mail server or firewall documentation to see whether this is an option.

If full-fledged header rewriting is not available, you still might prevent the sending of some critical information, such as server software version numbers and internal IP addresses.

X-Apparently-To:	my~secret~account!@yahoo.com via someone_else's_ip_address; Wed, 04 Feb 2004 09:39:49 -0800
Return-Path:	<kbeaver@principlelogic.com>
Received:	from someone_else's_ip_address (EHLO ISP_email_server) (someone_else's_ip_address) by Yahoo_email_server with SMTP; Wed, 04 Feb 2004 09:39:48 -0800
Received:	from my_email_server ([ip_address]) by ISP_email_server (IntelliMail vM.5.01.06.05 201-253-122-130-105-20030824) with ESMTP id <20040204173942.FYWC1950.ISP_email_server@my_email_server> for <my~secret~account!@yahoo.com>; Wed, 4 Feb 2004 12:39:42 -0500
Received:	from MY HOST NAME (Not Verified[10.11.12.211]) by my_email_server with Generic SMTP Server v1.0a id <B00000f611>; Wed, 04 Feb 2004 12:39:35 -0500
Message-ID:	<000801c3eb46$258927a0$8001010f >
From:	"Kevin Beaver" <kbeaver@principlelogic.com> Add to Address Book
To:	my~secret~account!@yahoo.com
Subject:	See my headers?
Date:	Wed, 4 Feb 2004 12:40:38 -0500
MIME-Version:	1.0
Content-Type:	multipart/alternative; boundary="----=_NextPart_000_0005_01C3EB1C.1762FA00"
X-Priority:	3
X-MSMail-Priority:	Normal
X-Mailer:	Microsoft Outlook Express 6.00.2800.1158
X-MimeOLE:	Produced By Microsoft MimeOLE V6.00.2800.1165
Content-Length:	661

Figure 13-10: Critical information revealed in e-mail headers.

Capturing traffic

E-mail traffic, including usernames and passwords, can be captured with a network analyzer or an e-mail packet sniffer and reconstructor.

Mailsnarf is an e-mail packet sniffer and reconstructor that's part of the dsniff package (`www.monkey.org/~dugsong/dsniff`). There's a great commercial (yet low-cost) program called NetResident (`www.tamos.com/products/netresident`), too. You can also use Cain & Abel (`www.oxid.it/cain.html`) to highlight e-mail-in-transit weaknesses. I cover password cracking using this tool and others in Chapter 7.

If traffic is captured, a hacker or malicious insider can compromise one host and potentially have full access to another adjacent host, such as your e-mail server.

Malware

E-mail systems are regularly attacked by such malware as viruses and worms. One of the most important tests you can run for malware vulnerability is to verify that your antivirus software is actually working.

Before you begin testing your antivirus software, make sure that you have the latest virus software engine and signatures loaded.

You have a couple of safe options for checking the effectiveness of your antivirus software, as described in the following two sections. This is by no means a comprehensive method of testing for malware vulnerabilities, but it serves as a good, safe start.

EICAR test string

EICAR is a European-based malware "think tank" that has worked in conjunction with anti-malware vendors to provide this basic system test. The EICAR test string transmits in the body of an e-mail or as a file attachment so that you can see how your server and workstations respond. You basically access this file — which contains the following 68-character string — on your computer to see whether your antivirus or other malware software detects it:

```
X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR STANDARD-ANTIVIRUS-TEST-FILE!$H+H*
```

You can download a text file with this string from `www.eicar.org/anti_virus_test_file.htm`. Several versions of the file are available on this site. I recommend testing with the Zip file to make sure that your antivirus software can detect malware within compressed files.

When you run this test, you may see results from your antivirus software similar to Figure 13-11.

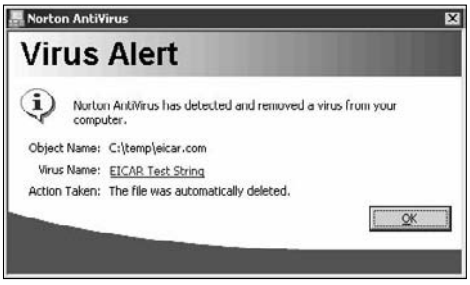


Figure 13-11: Using the EICAR test string to test antivirus software.

GFI's Email Security Testing Zone

A freebie at `www.gfi.com/emailsecuritytest` is a good e-mail malware test to run against your server and clients. This tool performs more than a dozen checks by sending e-mails containing the EICAR test file, malicious-like scripts, and malformed messages to check exactly what can get through and what can be exploited on your e-mail system. These aren't malicious tests — just tests that *should* invoke your antivirus software or other protective measures on your e-mail server or gateway if your software is configured and working correctly.

E-mail systems can be attacked using other tools I cover in this book, such as Metasploit (`www.metasploit.com`) for exploiting missing patches in Exchange and other servers, and Brutus (`www.hoobie.net/brutus`) for cracking POP3 passwords.

General best practices for minimizing e-mail security risks

The following countermeasures help keep messages as secure as possible.

Software solutions

The right software can neutralize many threats:

- **Use malware-protection software on the e-mail server —** better, the e-mail gateway — to prevent malware from reaching e-mail clients.
- **Apply the latest operating system and e-mail application security patches consistently and after any security alerts are released.**
- **If it makes good business sense, encrypt messages.** You can use S/MIME or PGP to encrypt sensitive messages or use e-mail encryption at the desktop level or the server or e-mail gateway. You can also use SSL/TLS via the POP3S, IMAPS, and SMPTS protocols.

  Do not depend on your users to encrypt messages. Use an enterprise solution to encrypt messages automatically instead.

  Make sure that encrypted files and e-mails can be protected against malware.

  - Encryption doesn't keep malware out of files or e-mails. You just have encrypted malware within the files or e-mails.
  - Encryption keeps your server or gateway antivirus from detecting the malware until it reaches the desktop.
- **Make it policy for users not to open unsolicited e-mails or any attachments,** especially those from unknown senders, and create ongoing awareness sessions and other reminders.
- **Plan for users who ignore or forget about the policy of leaving unsolicited e-mails and attachments unopened. It will happen!**

Operating guidelines

Some simple operating rules can keep your walls high and the attackers out of your e-mail systems:

- Put your e-mail server behind a firewall on a different network segment from the Internet and from your internal LAN — ideally in a demilitarized zone (DMZ).
- Disable unused protocols and services on your e-mail server.
- Run your e-mail server on a dedicated server, if possible, to help keep malicious attacks out of other servers and information if the e-mail server is hacked.

- Log all transactions with the server in case you need to investigate malicious use.
- If your server doesn't need certain e-mail services running (SMTP, POP3, and IMAP), disable them — immediately.
- For Web-based e-mail, such as Microsoft's Outlook Web Access (OWA), properly test and secure your Web server application and operating system by using the testing techniques and hardening resources I mention throughout this book.
- Require strong passwords. I cover password hacking in Chapter 7.
- If you're running sendmail — especially an older version — consider running a secure alternative, such as Postfix or qmail.

Instant Messaging

Instant messaging (IM) is yet another application that's catching many administrators off guard. Although IM offers a lot of business value, some serious security issues are associated with it. This is especially true if the application isn't managed properly and end users are free to install, configure, and use it in any way they want.

IM vulnerabilities

IM has several critical security vulnerabilities, including:

- Name hijacking, allowing a hacker to assume the identity of an IM user
- Exploiting a vulnerability in the IM client, allowing the attacker to take remote control of the computer
- Transferring malware, including viruses and malicious Trojan horses

You can remedy most of these vulnerabilities by applying the latest software patches and keeping antivirus signatures up to date. However, two IM vulnerabilities are susceptible to malicious attack, so they deserve a little more discussion. These vulnerabilities — problems with file sharing and log files — affect most of the popular IM clients, including AOL Instant Messenger (AIM) and ICQ, but shouldn't be ignored when securing your network.

Sharing files

The biggest problem I see with IM clients is their ability to share files, which can lead to data leakage. This feature might be neat for home users or others with standalone computers, but it can pose a real security risk to your business. Practically every IM client gives users the ability to share both local and network files.

The best way to determine IM usage on your network is to use a network analyzer and monitor IM traffic. For instance, you can use Wireshark (`www.wireshark.org`) or NetResident (`www.tamos.com/products/netresident`) to capture and display various types of IM protocols, such as AOL Instant Messenger, ICQ, MSN Messenger, and Jabber, as shown in Figure 13-12.

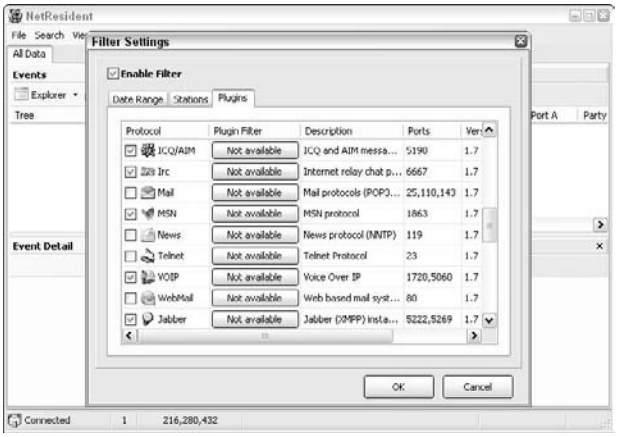


Figure 13-12: NetResident for monitoring IM traffic on your network.

Log files

Many IM clients can log all IM conversations. Some clients log all conversations by default. Have users enabled logging and inadvertently shared their log files with the world? For example, the log shown in Figure 13-13 is a smoking gun for a hacker — or the District Attorney — to use! Figure 13-13 shows part of an ICQ conversation stored in communications gobbledygook in a log file found in the `c:\Program Files\ICQ` folder.

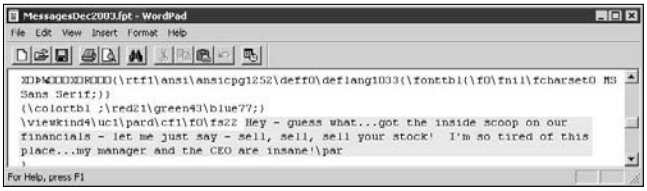


Figure 13-13: IM log files revealing juicy information.

Countermeasures against IM vulnerabilities

IM vulnerabilities can be difficult to detect because IM software is workstation-based. If you have a large network, checking every computer

for these vulnerabilities is impossible. Spot checks are inaccurate because every desktop and every user can be different.

Even if you disallow IM — or other messaging software — on your network, users will almost always find a way around your policy. However, if you implement these countermeasures, you're better prepared to protect your users from themselves and from hackers.

Detecting IM traffic

In addition to a network analyzer, you can detect IM traffic by using the following tools:

- Quest Policy Authority for UC (`www.quest.com`) and FaceTime's IMAuditor (`www.facetime.com`). If you can justify the cost — which is relatively easy — I recommend that you check out these products.
- Desktop auditing utilities can show you which applications are installed, including their specific settings. Such products as Ecora's Auditor Professional (`www.ecora.com/ecora/products/enterprise_auditor.asp`) and Microsoft's System Center ConfigurationManager (`www.microsoft.com/smserver/default.mspx`) also offer this functionality.

Maintenance and configuration

In addition to using the tools listed in the previous section, you can implement these IM hacking countermeasures:

- **User behavior:**
  - Have a policy banning or limiting the usage of all P2P software, including IM, BitTorrent, Google Talk, and so on.
  - Instruct users not to open file attachments or configure their IM software to share or receive file attachments.
  - Instruct users to keep their buddy lists private and not to share their information.
- **System configuration:**
  - Change default IM software installation directories to help eliminate automated attacks.
  - Apply all the latest IM software patches.
  - Ensure that the latest antivirus software and personal firewall software is loaded on each IM client.
  - Ensure that proper file and directory access controls are in place to give your users the minimum necessary rights for their jobs. This countermeasure helps keep prying eyes out when someone exploits an IM vulnerability.

- If you allow IM on your network for business purposes, consider standardizing on an enterprise-based IM application, such as Jabber or Lotus Sametime. These applications have more robust and manageable security options, which can ensure control.

Voice over IP

The hottest new technology blowing through town these days is undoubtedly Voice over IP (VoIP). Whether it's in-house VoIP systems or systems for remote users, VoIP servers, soft phones, and other related components have a slew of vulnerabilities. Like most things security related, many people haven't thought about the security issues surrounding voice conversations traversing their networks or the Internet — but it certainly needs to be on your radar. Don't fret — it's not too late to make things right, especially since VoIP is still relatively young. Just remember, though, that even if protective measures are in place, VoIP systems need to be included as part of your overall ethical hacking strategy on a continuous basis.

VoIP vulnerabilities

As with any new technology or set of network protocols, the bad guys are always going to figure out how to break in. VoIP is certainly no different. In fact, given what's at stake (phone conversations and phone system availability), there's certainly a lot to lose.

VoIP-related systems are no more secure than other common computer systems. Why? It's simple. VoIP systems have their own operating system, they have IP addresses, and they're accessible on the network. Compounding the issue is the fact that many VoIP systems house more *intelligence* — a fancy word for "more stuff that can go wrong" — which makes VoIP networks even more hackable.

If you want to find out more about how VoIP operates, which will undoubtedly help you root out vulnerabilities, check out *VoIP For Dummies* by Timothy V. Kelly.

On one hand, VoIP systems have vulnerabilities very similar to other systems I cover in this book, including:

- Default settings
- Missing patches
- Weak passwords

That's why using the standard vulnerability scanning tools I cover is important. Figure 13-14 shows various vulnerabilities associated with the authentication mechanism in the Web interface of a VoIP adapter.

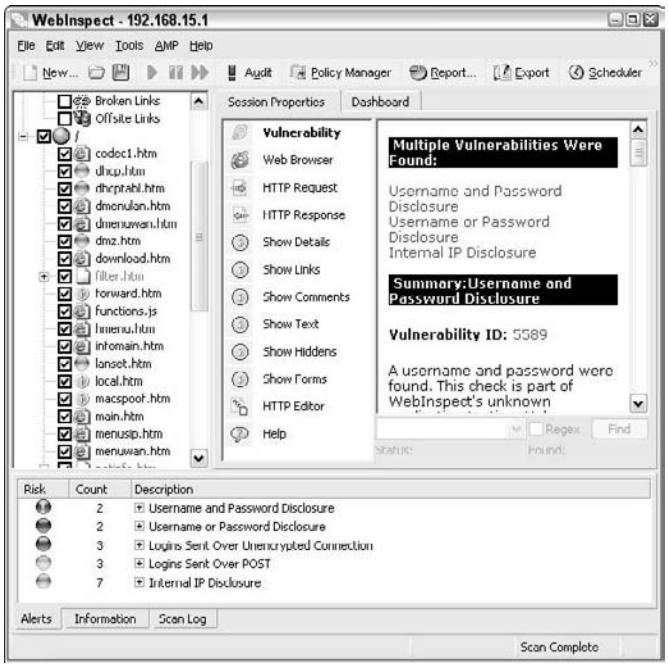


Figure 13-14: WebInspect scan of a VoIP network adapter showing several weaknesses.

Looking at these results, apparently this device is just a basic Web server. That's exactly my point — VoIP systems are nothing more than networked computer systems that have vulnerabilities that can be exploited.

On the other hand, two major security weaknesses are tied specifically to VoIP. The first is that of phone service disruption. Yep, VoIP is susceptible to denial of service just like any other system or application. VoIP is as vulnerable as the most timing-sensitive applications out there, given the low tolerance folks have for choppy and dropped phone conversations (cell phones aside, of course). The other big weakness with VoIP is that voice conversations are not encrypted thus can be intercepted and recorded. Imagine the fun a bad guy could have recording conversations and blackmailing his victims. This is very easy on unsecured wireless networks, but, as I show in the upcoming "Capturing and recording voice traffic" section, it's also pretty simple to carry out on wired networks, too.

If a VoIP network is not protected via network segmentation, such as a virtual local area network (VLAN), then the voice network is especially susceptible to eavesdropping, denial of service, and other attacks. But the VLAN barrier can be overcome in Cisco, Avaya, and Nortel environments by using a tool called VoIP Hopper (`http://voiphopper.sourceforge.net`). Just when

you think your voice systems are secure, a tool like VoIP Hopper comes along. Gotta love innovation!

Unlike typical computer security vulnerabilities, these issues with VoIP aren't easily fixed with simple software patches. These vulnerabilities are embedded into the Session Initiation Protocol (SIP) and Real-time Transport Protocol (RTP) that VoIP uses for its communications. The following are two VoIP-centric tests you should use to assess the security of your voice systems.

Scanning for vulnerabilities

Outside the basic network, OS, and Web application vulnerabilities, you can uncover other VoIP issues if you use the right tools. A neat Windows-based tool that's dedicated to finding vulnerabilities in VoIP networks is SiVuS (`http://vopsec.net/html/tools.html`). SiVuS allows you to perform the basic ethical hacking steps of scanning, enumerating, and rooting out vulnerabilities. You can start by downloading and running the SiVuS installation executable (currently v1.09).

After SiVuS is installed, load the program and you're ready to get started. Figure 13-15 shows my results of the first SiVuS step — Component Discovery.

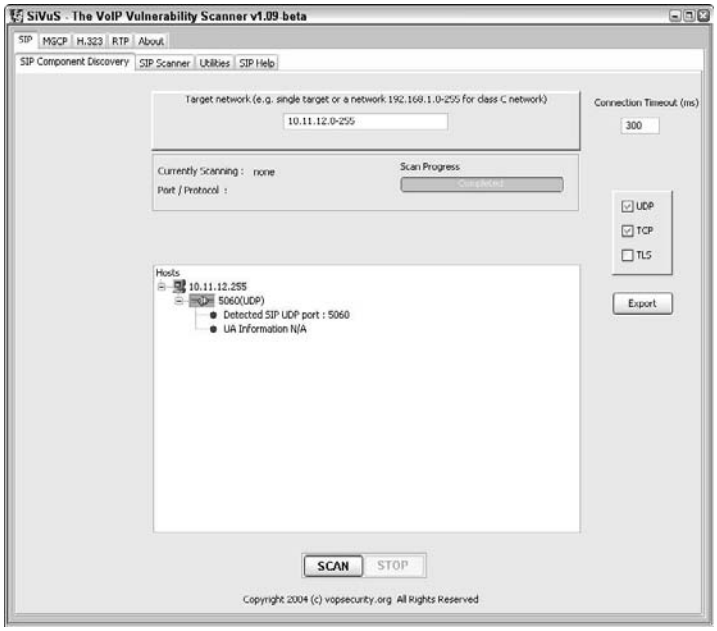


Figure 13-15: Using SiVuS Component Discovery to find live VoIP systems on the network.

You can use Component Discovery to search for one or two specific VoIP hosts, or you can scan your entire network. I recommend the latter because I find looking for one specific host is a little quirky — and you never know what other VoIP systems are out there that you could overlook.

After you find a few hosts, you can use SiVuS to dig deeper and root out DoS, buffer overflow, weak authentication, and other vulnerabilities related to VoIP. You can test each of your VoIP hosts for these vulnerabilities by using the following steps:

1. **Click the SIP Scanner tab and then click the Scanner Configuration tab.**
2. **In the Target(s) field in the upper-left corner, enter the system(s) you wish to scan and leave all other options at their defaults.**

   At this point, you can save the current configuration by clicking Save Configuration in the lower-right corner of the window. This creates a template you can use for your other hosts so that you don't have to change your settings each time.
3. **Click the Scanner Control Panel tab and either leave the default configuration or select your custom configuration in the Current Configuration drop-down list.**
4. **Click the green Scan button to start your scan.**
5. **When SiVuS finishes its tests, you hear a busy signal (assuming you have a sound card) signifying that testing is complete.**

   Your results might look similar to the SiVuS output shown in Figure 13-16.

Whether SiVuS's results and recommendations are an issue in your environment, I encourage you to sift through each one to determine what can and should be fixed. Remember, odds are that the bad guys both inside and outside your network can see these vulnerabilities just as easily as you can.

You can also use SiVuS to generate SIP messages, which come in handy if you want to test any built-in VoIP authentication mechanisms on your VoIP hosts. SiVuS's documentation (`www.vopsec.net/SiVuS-User-Doc.pdf`) outlines the specifics.

Other free tools for analyzing SIP traffic are PROTOS (`www.ee.oulu.fi/research/ouspg/protos/testing/c07/sip/index.html`), and sipsak (`http://sipsak.org`). A good Web site that lists all sorts of VoIP tools is `www.voipsa.org/Resources/tools.php`.

Figure 13-16: SiVuS discovered several VoIP-centric vulnerabilities.

Capturing and recording voice traffic

If you have access to the wired or wireless network, you can capture VoIP conversations easily. This is a great way to prove that the network and the VoIP installation are vulnerable. There are many legal issues associated with tapping into phone conversations, so make sure you have permission and are careful not to abuse your test results.

You can use Cain & Abel (technically just Cain for the features I demonstrate here) to tap into VoIP conversations. You can download Cain & Abel free at `www.oxid.it/cain.html`. Using Cain's ARP poison routing feature, you can plug in to the network and have it capture VoIP traffic:

1. **Load Cain & Abel and then click the Sniffer tab to enter the network analyzer mode.**

   The Hosts page opens by default.

2. **Click the Start/Stop APR icon (it looks like the nuclear waste symbol).**

   The ARP poison routing process starts and enables the built-in sniffer.

3. **Click the blue + icon to add hosts to perform ARP poisoning on.**

4. **In the MAC Address Scanner window that appears, ensure that All Hosts in My Subnet is selected and then click OK.**

5. **Click the APR tab (the one with the yellow-and-black circle icon) to load the APR page.**

6. **Click the white space under the uppermost Status column heading (just under the Sniffer tab).**

   This re-enables the blue + icon.

7. **Click the blue + icon and the New ARP Poison Routing window shows the hosts discovered in Step 3.**

8. **Select your default route or other host that you want to capture packets traveling to and from.**

   I just select my default route, but you might consider selecting your SIP manager or other central VoIP system. The right column fills with all the remaining hosts.

9. **In the right column, Ctrl+click the system you want to poison to capture its voice traffic.**

   In my case, I select my VoIP network adapter, but you might consider selecting all your VoIP phones.

10. **Click OK to start the ARP poisoning process.**

    This process can take anywhere from a few seconds to a few minutes depending on your network hardware and each host's local TCP/IP stack.

11. **Click the VoIP tab and all voice conversations are "automagically" recorded.**

    Here's the interesting part — the conversations are saved in `.wav` audio file format, so you simply right-click the recorded conversation you want to test and choose Play, as shown in Figure 13-17. Note that conversations being recorded show *Recording...* in the Status column.

The voice quality with Cain and other tools depends on the codec your VoIP devices use. With my equipment, I find the quality is marginal at best. That's not really a big deal, though, because your goal is to prove there's a vulnerability — not to listen in on other people's conversations.

There's also a Linux-based tool called vomit (`http://vomit.xtdnet.nl`) — short for voice over misconfigured Internet telephones — that you can use to convert VoIP conversations into `.wav` files. You first need to capture the actual conversation by using tcpdump, but if Linux is your preference, this solution offers basically the same results as Cain, outlined in the preceding steps.

If you're going to work a lot with VoIP, I highly recommend you invest in a good VoIP network analyzer. Check out WildPackets' OmniPeek — a great all-in-one wired and wireless analyzer (`www.wildpackets.com/products/distributed_network_analysis/omnipeek_network_analyzer`) — and TamoSoft's CommView (`www.tamos.com/products/commview`), which is a great low-priced alternative.

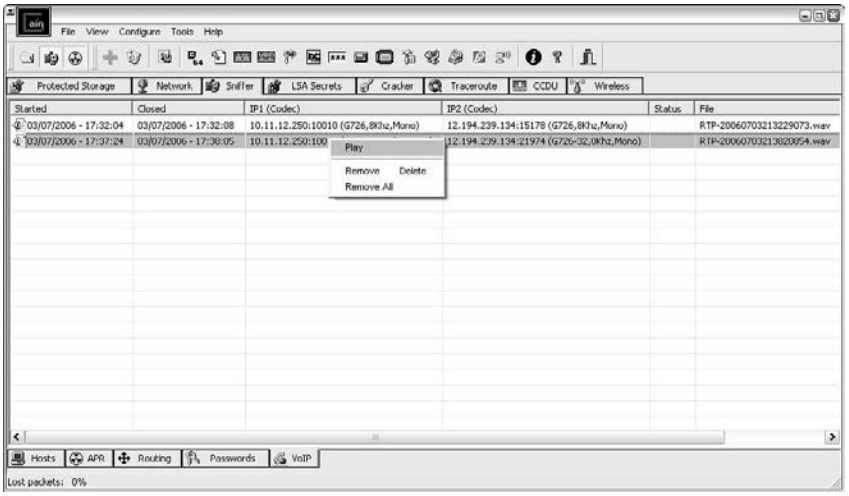


Figure 13-17: Using Cain & Abel to capture and record VoIP conversations.

These VoIP vulnerabilities are only the tip of the iceberg. New systems, software, and related protocols continue to emerge, so it pays to remain vigilant, helping to ensure your conversations are "locked down" from those with malicious intent.

Countermeasures against VoIP vulnerabilities

Locking down VoIP can be tricky. You can get off to a good start, though, by segmenting your voice network into its own VLAN — or even a dedicated physical network if that fits into your budget. You should also make sure that all VoIP-related systems are hardened according to vendor recommendations and widely accepted best practices (such as NIST's SP800-58 document at `http://csrc.nist.gov/publications/nistpubs/800-58/SP800-58-final.pdf`) and that software and firmware are fully patched.

Chapter 14

Web Sites and Applications

In This Chapter

- Testing Web applications
- Hacking with Google
- Protecting against SQL injection and cross-site scripting
- Preventing login weaknesses
- Countering Web application abuse
- Analyzing the source code

Web applications are common targets for attack because they're everywhere and often open for anyone to poke and prod. Basic Web sites used for marketing, contact information, document downloads, and so on are common targets for the bad guys to play around with (especially the script-kiddie types). However, for criminal hackers, Web sites that provide a front end to complex applications and databases that store valuable information, such as credit card and Social Security numbers, are especially attractive. This is where the money is, both literally and figuratively.

Why are Web sites and applications so vulnerable? The consensus is that they're vulnerable because of poor software development and testing practices. Sound familiar? It should; this same problem affects operating systems and practically all computer systems. This is the side effect of relying on software compilers to perform error checking, waning user demand for higher-quality software, and emphasizing time-to-market instead of security and stability.

This chapter presents Web site and application hacks to run on your systems. Given all the custom software configuration possibilities, you can test for literally thousands of Web vulnerabilities, but I focus on the ones I see most often using both automated scanners and manual analysis. Just know that what's contained in this chapter only scratches the surface of Web hacking possibilities. I also outline countermeasures to help minimize the chances that a hacker can carry out these attacks against what are likely considered your most critical systems.

Choosing Your Web Application Tools

Good Web vulnerability scanners and tools can help ensure that you get the most from your scans. As with most things in life, I find that you get what you pay for when it comes to testing for Web security holes. This is why I mostly use commercial tools in my work testing Web sites and applications for vulnerabilities.

These are my favorite Web application testing tools:

- **Acunetix Web Vulnerability Scanner** (`www.acunetix.com`) for all-in-one security testing, including a port scanner, an HTTP sniffer, and an automated SQL injection tool
- **Firefox Web Developer** (`http://chrispederick.com/work/web-developer`) for manual analysis and manipulation of Web pages

Web hacking involves much more than just running automated scanning tools. They find around half of the issues, but you have to pick up where they leave off to truly assess the overall Web site and application. This isn't a fault of Web vulnerability scanners but rather the nature of the beast. Poking and prodding Web sites and applications requires good old-fashioned hacker trickery and your favorite Web browser.

- **HTTrack Website Copier** (`www.httrack.com`) for mirroring a site for offline inspection

Mirroring is a method for crawling through (also called *spidering*) a Web site's every nook and cranny and downloading publicly accessible pages to your local system.

- **N-Stalker Web Application Security Scanner** (`www.nstalker.com/eng/products/nstealth`) for all-in-one security testing, including password cracking and Web server load testing tools
- **WebInspect** (`www.spidynamics.com/products/webinspect/index.html`) for all-in-one security testing, including an excellent HTTP proxy and HTTP editor and an automated SQL injection tool

You can also use general vulnerability scanners, such as QualysGuard and LANguard, as well as exploit tools, such as Metasploit, when testing Web servers and applications. These tools can be used to find (and exploit) weaknesses at the Web server level that you might not otherwise find with standard Web scanning tools and manual analysis. Google can be beneficial for rooting through Web applications and looking for sensitive information. Although these non–application-specific tools can be beneficial, it's important to know that they won't drill down as deep as the tools given in the preceding list.

Case study in hacking Web applications with Caleb Sima

In this case study, Caleb Sima, a well-known application security expert, shared an experience of performing a Web-application security test.

The Situation

Mr. Sima was hired to perform a Web application penetration test to assess the security of a well-known financial Web site. Equipped with nothing more than the URL of the main financial site, Mr. Sima set out to find what other sites existed for the organization and began by using Google to search for possibilities. Mr. Sima initially ran an automated scan against the main servers to discover any low-hanging fruit. This scan provided information on the Web server version and some other basic information, but nothing that proved useful without further research. And while Mr. Sima performed the scan, neither the IDS nor the firewall noticed any of his activity. Then Mr. Sima issued a request to the server on the initial Web page, which returned some interesting information. The Web application appeared to be accepting many parameters, but as Mr. Sima continued to browse the site, he noticed that the parameters in the URL stayed the same. Mr. Sima decided to delete all the parameters within the URL to see what information the server would return when queried. The server responded with an error message describing the type of application environment.

Next, Mr. Sima performed a Google search on the application that resulted in some detailed documentation. Mr. Sima found several articles and tech notes within this information that showed him how the application worked and what default files might exist. In fact, the server had several of these default files. Mr. Sima used this information to probe the application further. He quickly discovered internal IP addresses and what services the application was offering. Now that Mr. Sima knew exactly what version the admin was running, he wanted to see what else he could find.

Mr. Sima continued to manipulate the URL from the application by adding `&` characters within the statement to control the custom script. This allowed him to capture all source code files. Mr. Sima noted some interesting filenames, including `VerifyLogin.htm`, `ApplicationDetail.htm`, `CreditReport.htm`, and `ChangePassword.htm`. Then Mr. Sima tried to connect to each file by issuing a specially formatted URL to the server. The server returned a *User not logged in* message for each request and stated that the connection must be made from the intranet.

The Outcome

Mr. Sima knew where the files were located and was able to sniff the connection and determine that the `ApplicationDetail.htm` file set a cookie string. With little manipulation of the URL, Mr. Sima hit the jackpot. This file returned client information and credit cards when a new customer application was being processed. `CreditReport.htm` allowed Mr. Sima to view customer credit report status, fraud information, declined-application status, and a multitude of other sensitive information. The lesson: Hackers can utilize many types of information to break through Web applications. The individual exploits in this case study were minor, but when combined, they resulted in severe vulnerabilities.

Caleb Sima was a charter member of the X-Force team at Internet Security Systems and was the first member of the penetration testing team. Mr. Sima went on to co-found SPI Dynamics (later acquired by HP) and become its CTO, as well as director of SPI Labs, the application-security research and development group within SPI Dynamics.

Web Vulnerabilities

Attacks against unsecured Web sites and applications via Hypertext Transfer Protocol (HTTP) make up the majority of all Internet-related attacks. Most of these attacks can be carried out even if the HTTP traffic is encrypted (via HTTPS, or HTTP over SSL) because the communications medium has nothing to do with these attacks. The security vulnerabilities actually lie within the Web sites and applications themselves or the Web server and browser software that the systems run on and communicate with.

Many attacks against Web sites and applications are just minor nuisances and might not affect sensitive information or system availability. However, some attacks can wreak havoc on your systems putting sensitive information at risk and even placing your organization out of compliance with state, federal, and international privacy and security laws and regulations.

Directory traversal

Let's start with a simple directory traversal attack. Directory traversal is a really basic weakness, but it can turn up interesting — sometimes sensitive — information about a Web system. This attack involves browsing a site and looking for clues about the server's directory structure and sensitive files that might have been loaded intentionally or unintentionally.

Perform the following tests to determine information about your Web site's directory structure.

Crawlers

A spider program, such as the free HTTrack Website Copier, can crawl your site to look for every publicly accessible file. To use HTTrack, simply load it, give your project a name, tell HTTrack which Web site(s) to mirror, and after a few minutes (depending on the size and complexity of the site), you'll have everything that's publicly accessible on the site stored on your local drive in `c:\My Web Sites`. Figure 14-1 shows the crawl output of a basic Web site.

Complicated sites often reveal more information that should not be there, including old data files and even application scripts and source code.

During a recent Web security assessment project, I stumbled across a `.zip` file in a Web server's download directory. When I tried to open the file, the system asked me for a password. Using my handy dandy Zip password cracking tool (see Chapter 7 for details on password cracking), I had the password in mere milliseconds. Inside the Zip file was an Excel spreadsheet containing sensitive patient healthcare information (names, addresses, Social Security numbers, and more) that anyone and everyone in the world could access.

In situations like this, your business might be required to notify everyone involved that their information was unprotected and possibly compromised. It pays to know the laws and regulations affecting your business. Better yet, make sure users aren't posting sensitive information in the first place!

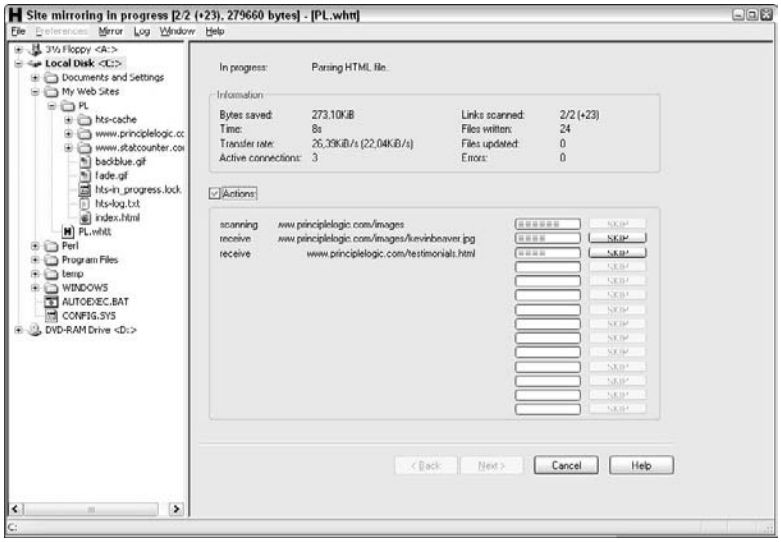


Figure 14-1: Using HTTrack to crawl a Web site.

Look at the output of your crawling program to see what files are available. Regular HTML and PDF files are probably okay because they're most likely needed for normal Web usage. But it wouldn't hurt to open each file to make sure it belongs there and doesn't contain sensitive information you don't want to share with the world.

Google

Google, the search engine company that many love to hate, can also be used for directory traversal. In fact, Google's advanced queries are so powerful that you can use them to root out sensitive information, critical Web server files and directories, credit card numbers, Webcams — basically anything that Google has discovered on your site — without having to mirror your site and sift through everything manually.

The following are a couple of advanced Google queries that you can enter directly into the Google search field:

- *site:hostname keywords:* This will search for any keyword you list, such as *SSN, confidential, credit card,* and so on.
- *filetype:file-extension hostname:* This will search for specific file types, such as `.doc`, `.pdf`, `.db`, `.dbf`, `.zip`, and more, that might contain sensitive information.

Other advanced Google operators include the following:

- *allintitle* searches for keywords in the title of a Web page.
- *inurl* searches for keywords in the URL of a Web page.
- *related* finds pages similar to this Web page.
- *link* shows other sites that link to this Web page.

Specific definitions and more can be found at `www.google.com/intl/en/help/operators.html`. Also, an excellent resource for Google hacking is Johnny Long's Google Hacking Database (GHDB) site `http://johnny.ihackstuff.com/ghdb`. Additional hacking-related Google queries can be found at `http://artkast.yak.net/81`.

When sifting through your site with Google, be sure to look for sensitive information about your servers, network, and organization in Google Groups (`http://groups.google.com`), which is the Usenet archive. I often find employee postings in newsgroups that reveal too much about the internal network and business systems. If you find something that doesn't need to be there, you can supposedly work with Google to have it edited or removed.

Looking at the big picture of Web security, Google hacking is pretty limited, but if you're really into it, check out Johnny Long's book, *Google Hacking for Penetration Testers* (Syngress).

Countermeasures against directory traversals

You can employ two main countermeasures against having files compromised via malicious directory traversals:

- **Don't store old, sensitive, or otherwise nonpublic files on your Web server.** The only files that should be in your `/htdocs` or `DocumentRoot` folder are those that are needed for the site to function properly. These files should not contain confidential information that you don't want the world to see.
- **Configure your `robots.txt` file to prevent search engines, such as Google, from crawling the more sensitive areas of your site.**
- **Ensure that your Web server is properly configured to allow public access to only those directories that are needed for the site to function.** Minimum privileges are key here, so provide access to only the files and directories needed for the Web application to perform properly.

Check your Web server's documentation for instructions on controlling public access. Depending on your Web server version, these access controls are set in

- The `httpd.conf` file and the `.htaccess` files for Apache (refer to `http://httpd.apache.org/docs/configuring.html` for more information)
- Internet Information Services Manager settings for Home Directory and Directory (IIS 5.1)
- Internet Information Services Manager settings for Home Directory and Virtual Directory (IIS 6.0)

The latest versions of these Web servers have good directory security by default so, if possible, make sure you're running the latest versions.

Finally, consider using a search engine honeypot, such as the Google Hack Honeypot (`http://ghh.sourceforge.net`), to see how the bad guys are working against your site and to keep them at bay.

Input filtering attacks

Web sites and applications are notorious for taking practically any type of input, mistakenly assuming that it's valid, and processing it further. Not validating input is one of the greatest mistakes that Web developers can make.

Several attacks that insert malformed data — often, too much at one time — can be run against a Web site or application, which can confuse or crash the system or make it divulge too much information to the attacker. Input attacks can also make it easy for the bad guys to glean sensitive information from the Web browsers of unsuspecting users.

Buffer overflows

One of the most serious input attacks is a buffer overflow that specifically targets input fields in Web applications.

For instance, a credit reporting application might authenticate users before they're allowed to submit data or pull reports. The login form uses the following code to grab user IDs with a maximum input of 12 characters, as denoted by the `maxsize` variable:

```
<form name="Webauthenticate" action="www.your_Web_app.com/
login.cgi" method="POST">
...
<input type="text" name="inputname" maxsize="12">
...
```

A typical login session would involve a valid login name of 12 characters or less. However, the `maxsize` variable can be changed to something huge, such as 100 or even 1,000. Then an attacker can enter bogus data in the login field. What happens next is anyone's call — the application might hang, overwrite other data in memory, or crash the server.

A simple way to manipulate such a variable is to step through the page submission by using a Web proxy, such as those built in to the commercial Web vulnerability scanners I mention or the free Paros Proxy (`www.parosproxy.org`).

Web proxies sit between your Web browser and the server you're testing and allow you to manipulate information sent to the server. To begin, you must configure your Web browser to use the local proxy of 127.0.0.1 on port 8080. In Firefox, this is accessible by choosing Tools➪Options; click Advanced, click the Network tab, click the Connection Settings button, and then select the Manual Proxy Configuration radio button. In Internet Explorer, choose Tools➪Internet Options; click the Connections tab, click the LAN Settings button, and then select the Use a Proxy Server for Your LAN check box.

All you have to do is change the field length of the variable before your browser submits the page, and it will be submitted using whatever length you give. You can also use the Firefox Web Developer to remove maximum form lengths defined in Web forms, as shown in Figure 14-2.

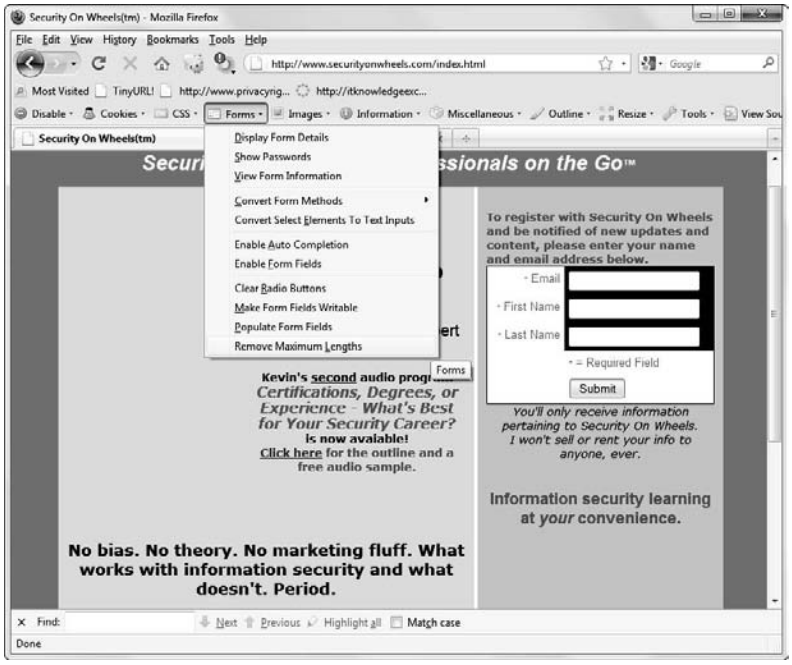


Figure 14-2: Using Firefox Web Developer to reset form field lengths.

URL manipulation

An automated input attack manipulates a URL and sends it back to the server, telling the Web application to do various things, such as redirect to third-party Web sites, load sensitive files off the server, and so on. Local file inclusion is one such vulnerability. This is when the Web application accepts URL-based input (often via CGI) and returns the specified file's contents to the user. For example, in one situation WebInspect sent something similar to the following request and returned the Linux server's passwd file:

```
https://www.your_Web_app.com/onlineserv/Checkout.
          cgi?state=
detail&language=english&imageSet=/../..//../..//../..//../
..///etc/passwd
```

The following links demonstrate another example of URL trickery called URL redirection:

```
http://www.your_Web_app.com/error.aspx?PURL=http://www.
bad~site.com&ERROR=Path+'OPTIONS'+is+forbidden.
http://www.your_Web_app.com/exit.asp?URL=http://www.
bad~site.com
```

In both situations, an attacker can exploit this vulnerability by sending the link to unsuspecting users via e-mail or by posting it on a Web site. When users click the link, they can be redirected to a malicious third-party site containing malware or inappropriate material.

If you have nothing but time on your hands, you might uncover these types of vulnerabilities manually. However, in the interest of time (and sanity), these attacks are best carried out by running a Web vulnerability scanner because they can detect the weakness and send hundreds and hundreds of URL iterations to the Web system very quickly.

Hidden field manipulation

Some Web applications embed hidden fields within Web pages to pass state information between the Web server and the browser. Hidden fields are represented in a Web form as `<input type="hidden">`. Because of poor coding practices, hidden fields often contain confidential information (such as product prices on an e-commerce site) that should be stored only in a back-end database. Users shouldn't see hidden fields — hence, the name — but the curious attacker can discover and exploit them with these steps:

1. **View the HTML source code.**

   To see the source code in Internet Explorer, choose Page➪View Source. In Firefox, choose View➪Page Source.

2. **Change the information stored in these fields.**

   For example, a malicious user might change the price from $100 to $10.

3. **Re-post the page back to the server.**

   This allows the attacker to obtain ill-gotten gains, such as a lower price on a Web purchase.

Using hidden fields for authentication (login) mechanisms can be especially dangerous. I once came across a multi-factor authentication intruder lockout process that relied on a hidden field to track the number of times the user attempted to login. This variable could be reset to zero for each login attempt and thus facilitate a scripted dictionary or brute-force login attack. It was somewhat ironic that the process to prevent intruder attacks was vulnerable to an intruder attack.

Several tools, such as Web Proxy (which comes with WebInspect) or Paros Proxy, can easily manipulate hidden fields. Figure 14-3 shows SPI Proxy's interface and a Web page's hidden field.

If you come across hidden fields, you can try to manipulate them to see what can be done. It's as simple as that.

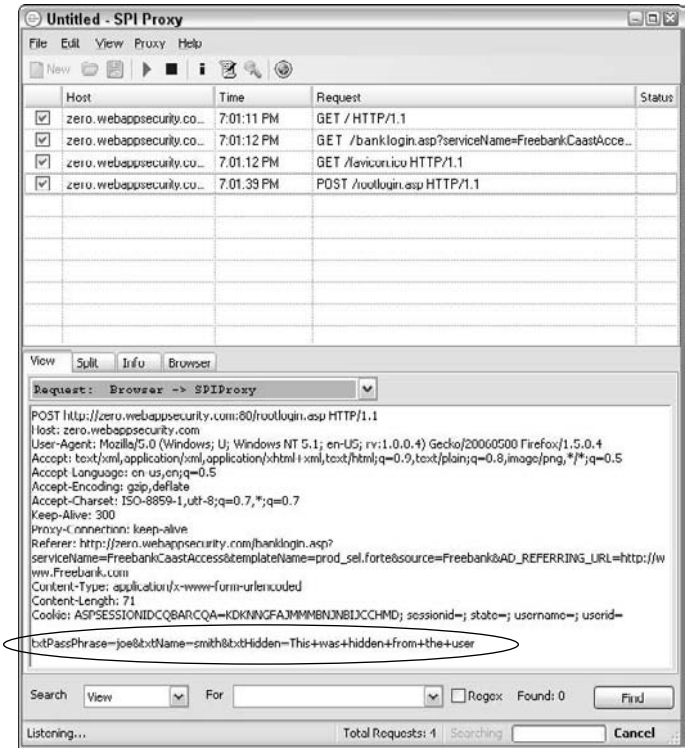


Figure 14-3: Using SPI Proxy to find and manipulate hidden fields.

Code injection and SQL injection

Similar to URL manipulation attacks, code-injection attacks manipulate specific system variables, for example:

```
http://www.your_Web_app.com/script.php?info_variable=X
```

Attackers, seeing this variable, can start entering different data into the `info_variable` field, changing `X` to something like one of the following lines:

```
http://www.your_Web_app.com/script.php?info_variable=Y
```

```
http://www.your_Web_app.com/script.php?info_
          variable=123XYZ
```

The Web application might respond in a way that gives attackers more information than they want, such as detailed errors or access into data fields they're not authorized to access. The invalid input might also cause the application or the server to hang. Similar to the case study earlier in the chapter, hackers can use this information to determine more about the Web application and its inner workings, which can ultimately lead to a serious system compromise.

If HTTP variables are passed in the URL and are easily accessible, it's only a matter of time before someone exploits your Web application.

I used a Web application to manage my personal information that did just this. Because a `"name"` parameter was part of the URL, anyone could gain access to other people's personal information by changing the `"name"` value. For example, if the URL included `"name=kbeaver"` a simple change to `"name=jsmith"` would bring up J. Smith's home address, Social Security number, and so on. Ouch. I alerted the system administrator of this vulnerability. After a few minutes of denial, he agreed that it was indeed a problem and proceeded to work with the developers to fix it.

Code injection can also be carried out against back-end SQL databases — an attack known as *SQL injection.* Malicious attackers insert SQL statements, such as CONNECT, SELECT, and UNION, into URL requests to attempt to connect and extract information from the SQL database that the Web application interacts with. SQL injection is made possible by applications not properly validating input combined with informative errors returned from database servers and Web servers. Two general types of SQL injection are standard (also called error-based) and blind. *Error-based* SQL injection is exploited based on error messages returned from the application when invalid information is input into the system. *Blind* SQL injection happens when error messages are disabled, requiring the hacker or automated tool to guess what the database is returning and how it's responding to injection attacks.

There's a quick way to determine whether your Web application is vulnerable to SQL injection. Simply enter a single apostrophe (') in your Web form fields or at the end of the URL. If a SQL error is returned, odds are SQL injection is present. As with URL manipulation, you're much better off running a Web vulnerability scanner to check for SQL injection. Figure 14-4 shows numerous SQL injection vulnerabilities discovered by the WebInspect vulnerability scanner.

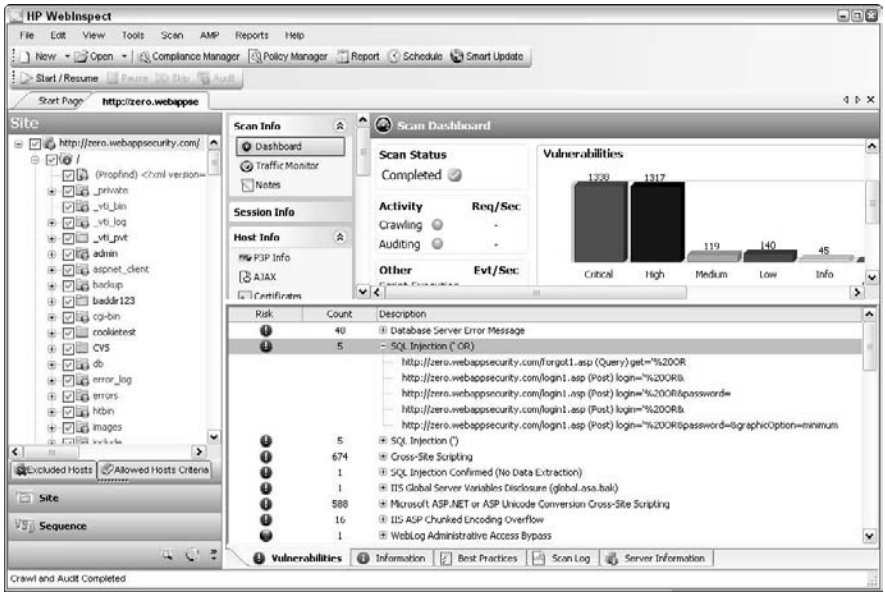


Figure 14-4: WebInspect discovered SQL injection vulnerabilities.

When you discover SQL injection vulnerabilities, you might be inclined to stop there. That's fine; however, I prefer to see how far I can get into the database system. An excellent — and amazingly simple — tool to use for this is SQL Injector, which comes with WebInspect. Acunetix Web Vulnerability Scanner has a similar tool built in as well. You simply provide the tool with the suspect URL that a scanner (such as Acunetix or WebInspect) discovered, select a few items, and you're connected to the database, as shown in Figure 14-5.

You can click the Get Data button in SQL Injector to start dumping information, as shown in Figure 14-6, leading you to the ultimate ethical hacking goal.

If your budget is limited, check out the free SQL injection tool called Absinthe (`www.0x90.org/releases/absinthe`).

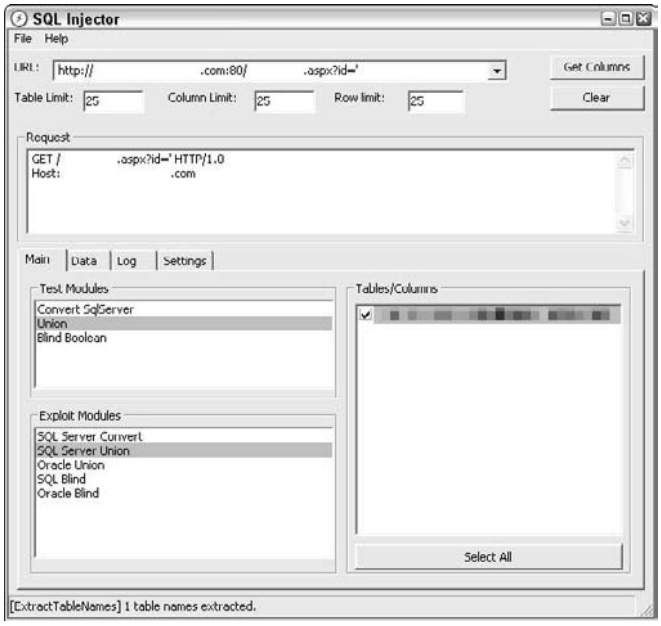


Figure 14-5: Using SQL Injector to connect to a SQL Server database table.

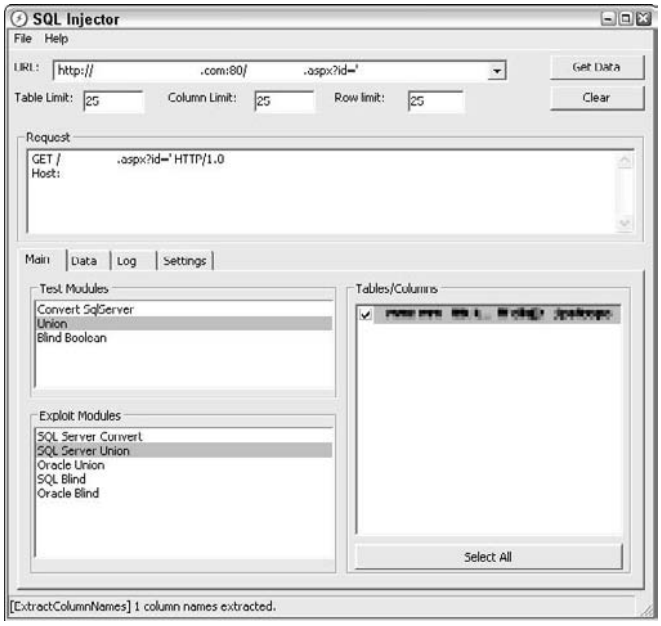


Figure 14-6: Using SQL Injector's Data Pump to extract column names.

I cover database security in depth in Chapter 15.

Cross-site scripting

Cross-site scripting (XSS) is perhaps the most well-known Web application vulnerability that occurs when a Web page displays user input — via JavaScript and VBScript — that isn't properly validated. A hacker can take advantage of the absence of input filtering and cause a Web page to execute malicious code on any user's computer that views the page.

For example, an XSS attack can display the user ID and password login page from another rogue Web site. If users unknowingly enter their user IDs and passwords in the login page, the user IDs and passwords are entered into the hacker's Web server log file. Other malicious code can be sent to a victim's computer and run with the same security privileges as the Web browser or e-mail application that's viewing it on the system; the malicious code could provide a hacker with full Read/Write access to browser cookies, browser history files, or even permit the download/installation of malware.

A simple test shows whether your Web application is vulnerable to XSS. Look for any fields in the application that accept user input (such as on a login or search form), and enter the following JavaScript statement:

```
<script>alert('XSS')</script>
```

If a window pops up that reads `XSS`, as shown in Figure 14-7, the application is vulnerable.

Figure 14-7: Script code reflected to the browser.

As with SQL injection, you really need to use an automated scanner to check for XSS. Both WebInspect and Acunetix Web Vulnerability Scanner do a great job of finding XSS. However, they often tend to find different XSS issues. This highlights that importance of using multiple scanners when you can. Figure 14-8 shows some sample XSS findings in Acunetix Web Vulnerability Scanner.

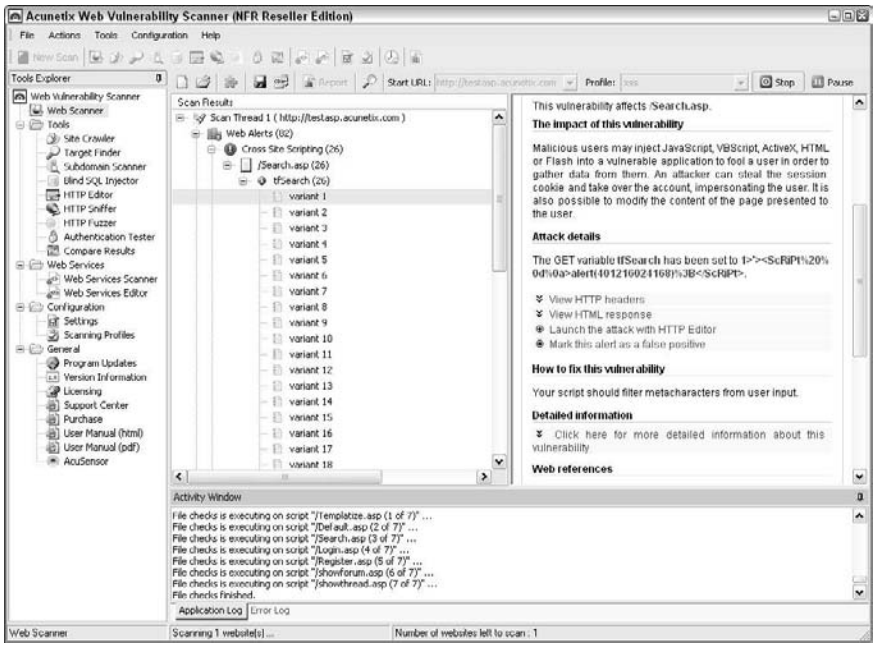


Figure 14-8: Using Acunetix Web Vulnerability Scanner to find cross-site scripting in a Web application.

Countermeasures against input attacks

Web applications must filter incoming data. The applications must check and ensure that the data entered fits within the parameters of what the application is expecting. If the data doesn't match, the application should generate an error or return to the previous page. Under no circumstances should the application accept the junk data, process it, and reflect it back to the user.

Secure software coding practices can eliminate all these issues if they're made a critical part of the development process. Developers should know and implement these best practices:

- Applications should never present static values that the Web browser and the user don't need to see. Instead, this data should be implemented within the Web application on the server side and retrieved from a database only when needed.
- Application should filter out `<script>` tags from input fields.
- Web server and database server error messages should be disabled if possible.

Sensitive information stored locally

Quite often as part of my ethical hacking, I use a hex editor to see how an application is storing sensitive information, such as passwords, in memory. When I'm in Firefox and Internet Explorer, I can use a hex editor, such as WinHex (`www.x-ways.net/winhex`), to search the active memory in each of these programs and frequently find user ID and password combinations.

I've found with Internet Explorer, this information is kept in memory even after browsing to several other Web sites or logging out of the application. This memory usage "feature" poses a security risk on the local system if another user accesses the computer, or if the system is infected with malware that can search system memory for sensitive information. The way browsers store sensitive information in memory is also bad news if an application error or system memory dump occurs and the user ends up sending the information to Microsoft (or other browser vendor) for QA purposes, or the information is written to a dump file on the local hard drive and sits there for someone to find.

Try this on your Web application(s) or standalone programs that require authentication. You just might be surprised at the outcome. Outside of obfuscating or encoding the login credentials, there's unfortunately not a great fix because this "feature" is part of the Web browser that developers can't really control.

A similar security feature occurs on the client side when HTTP GET requests rather than HTTP POST requests are used to process sensitive information. The following is an example of a vulnerable GET request:

```
https://www.your_Web_app.
    com/access.php?username=
    kbeaver&password=
    WhAteVur!&login=SoOn
```

GET requests are often stored in the user's Web browser history file, Web server log files, and proxy log files, and can be transmitted to third-party sites via the HTTP Referer field when the user browses to a third-party site. All of the above can lead to exposure of login credentials and unauthorized Web application access. The lesson: Don't use HTTP GET requests.

Default script attacks

Poorly written Web programs, such as Common Gateway Interface (CGI) and Active Server Pages (ASP) scripts, can allow hackers to view and manipulate files on a Web server and do other things they're not authorized to do. For example, the Poison Null Byte attack and the Upload Bombing attack against vulnerable CGI scripts written in Perl permit unauthorized access.

Default script attacks are common because so much poorly written code is freely accessible on Web sites. Hackers can also take advantage of various sample scripts that install on Web servers — especially older versions of Microsoft's IIS Web server.

Many Web developers and Webmasters use these scripts without understanding how they really work or without testing them, which can introduce serious security vulnerabilities.

To test for script vulnerabilities, you can peruse scripts manually or use a text search tool — such as the search function built in to the Windows Start menu or the Find program in Linux — to find any hard-coded usernames, passwords, and other sensitive information. Search for *admin, root, user, ID, login, signon, password, pass, pwd,* and so on. Sensitive information embedded in scripts like this is rarely necessary and is often the result of poor coding practices that give precedence to convenience over security.

A nice, low-cost tool for checking general Web application issues, such as script vulnerabilities and generating professional-looking reports, is N-Stalker Web Application Security Scanner, as shown in Figure 14-9.

Figure 14-9: Using N-Stalker Web Application Security Scanner to check a wide range of basic Web application vulnerabilities.

A free edition of N-Stalker is available at `http://nstalker.com/products/free`.

Countermeasures against default script attacks

You can help prevent attacks against default Web scripts as follows:

- Know how scripts work before deploying them within a Web application.
- Make sure that all default or sample scripts are removed from the Web server before using them.

  Don't use publicly accessible scripts that contain hard-coded confidential information. They're a security incident in the making.
- Set file permissions on sensitive areas of your site/application to prevent public access.

Unsecured login mechanisms

Many Web sites require users to log in before they can do anything with the application. These login mechanisms often do not handle incorrect user IDs or passwords gracefully. They often divulge too much information that an attacker can use to gather valid user IDs and passwords.

To test for unsecured login mechanisms, browse to your application and log in

- Using an invalid user ID with a valid password
- Using a valid user ID with an invalid password
- Using an invalid user ID and invalid password

After you enter this information, the Web application will probably respond with a message similar to `Your user ID is invalid` or `Your password is invalid`. The Web application might return a generic error message, such as `Your user ID and password combination is invalid` and, at the same time, return different error codes in the URL for invalid user IDs and invalid passwords, as shown in Figures 14-10 and 14-11.

In either case, this is bad news because the application is telling you not only which parameter is invalid, but also which one is *valid.* This means that malicious attackers now know a good username or password — their workload has been cut in half! If they know the username (which usually is easier to guess), they can simply write a script to automate the password-cracking process, and vice versa.

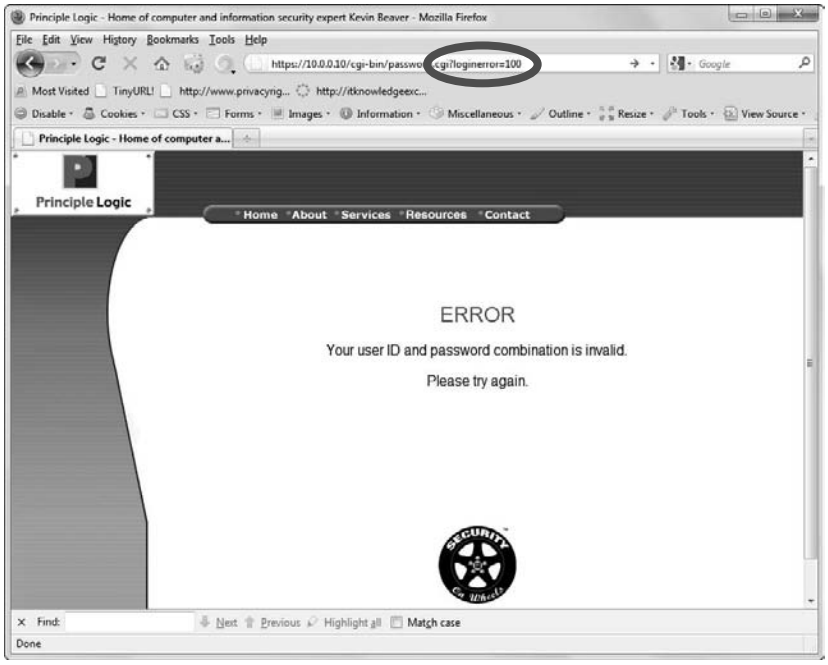


Figure 14-10: URL returns an error when an invalid user ID is entered.

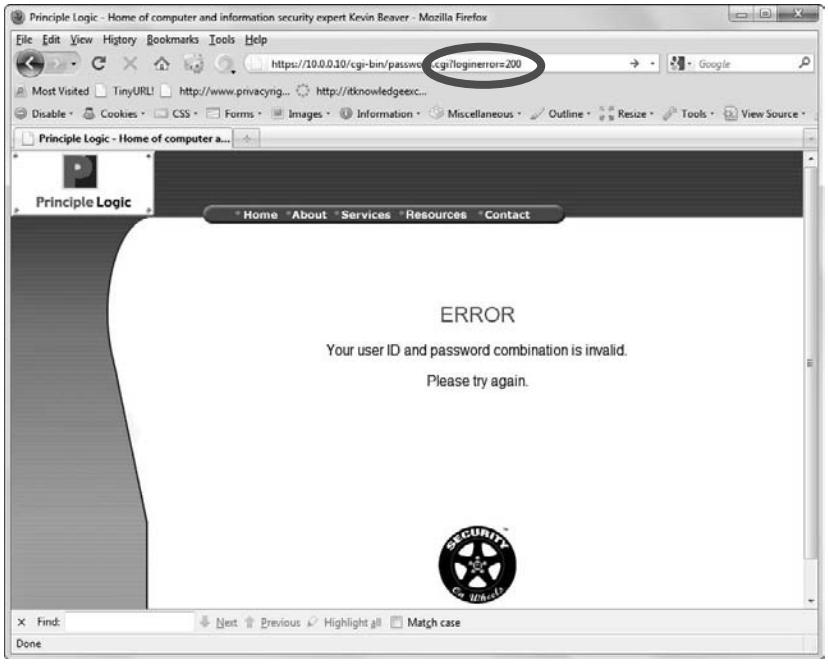


Figure 14-11: URL returns a different error when an invalid password is entered.

You should also take your login testing to the next level by using a Web login cracking tool, such as Brutus (`www.hoobie.net/brutus/index.html`), as shown in Figure 14-12. Brutus is a very simple tool that can be used to crack both HTTP and form-based authentication mechanisms by using both dictionary and brute-force attacks.

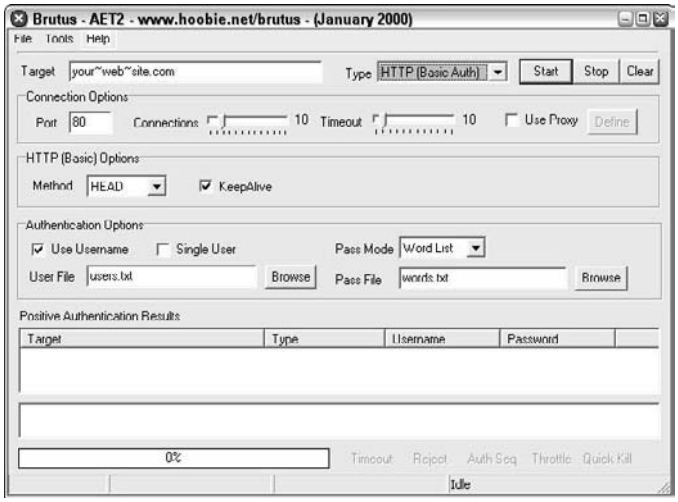


Figure 14-12: Brutus tool for testing for weak Web logins.

As with any type of password testing, this can be a long and arduous task, and you stand the risk of locking out user accounts. Proceed with caution.

Most commercial Web vulnerability scanners have decent dictionary-based Web password crackers but none (that I'm aware of) can do true brute-force testing like Brutus can. As I discuss in Chapter 7, your password-cracking success is highly dependent on your dictionary lists. I find the BlackKnightList (`http://rs159.rapidshare.com/files/184075601/BlackKnightList.rar`) is the most comprehensive.

Acunetix Web Vulnerability Scanner also tests for weak passwords during its scans. During a recent project, this scanner helped me find some weak Outlook Web Access (OWA) passwords that I wouldn't have found otherwise. This led to a much more in-depth security assessment of the OWA system I was testing and helped my client get a lot more bang for their buck.

You might not need a password cracking tool at all because many Web-based systems, such as printers and CCTV cameras, simply have the passwords that came on them — usually "password," "admin," or nothing at all.

Countermeasures against unsecured login systems

You can implement the following countermeasures to prevent people from attacking weak login systems in your Web applications:

- Any login errors that are returned to the end user should be as generic as possible, saying something similar to `Your user ID and password combination is invalid`.
- The application should never return error codes in the URL that differentiate between an invalid user ID and an invalid password.

  If a URL message must be returned, the application should keep it as generic as possible. Here's an example:

  ```
  www.your_Web_app.com/login.cgi?success=false
  ```

  This URL message might not be convenient to the user, but it helps hide the mechanism and the behind-the-scenes actions from the attacker.
- Use CAPTCHA (also reCAPTCHA) or Web login forms to prevent (or at least slow down) password cracking attempts.
- Employ an intruder lockout mechanism on your Web server or within your Web applications to lock user accounts after 10–15 failed login attempts. This can be handled via session tracking or via a third-party Web application firewall add-on like I discuss below.
- Change any vendor default passwords to something that's easy to remember yet difficult to crack.

General security scans for Web application vulnerabilities

I want to reiterate that both automated and manual testing need to be performed against your Web systems. You're not going to see the whole picture by relying on just one of these methods. I *highly* recommend using an all-in-one Web application vulnerability scanner, such as WebInspect, Acunetix Web Vulnerability Scanner, or N-Stalker Web Vulnerability Scanner to help you root out Web vulnerabilities that would be unreasonable if not impossible to find otherwise. Combine the scanner results with a malicious mindset and the hacking techniques I described in this chapter, and you're on your way to finding the Web security flaws that matter.

Hacking Web 2.0

Web 2.0 is changing how the Internet is used. From YouTube to Facebook to Twitter, new server and client-side technologies, such as Web Services, Ajax, and Flash, are being rolled out as if they're going out of style. And these aren't just consumer technologies. Businesses see the value in them and developers are excited to utilize the latest and greatest technologies in their environments.

Unfortunately, the downside to Web 2.0 is complexity. These new rich Internet applications (RIAs), as many call them, are so complex that developers, quality assurance analysts, and security managers are struggling to keep up with all their associated security issues. Don't get me wrong, the vulnerabilities in Web 2.0 applications are very similar to what we have with "legacy" technologies, such as XSS, SQL injection, parameter manipulation, and so on. The problem is that automated Web vulnerability scanners aren't quite mature enough — at least as of this writing — to find all the security weaknesses that count. When assessing the security of Web 2.0 applications, I find that most of them have to be analyzed manually. I'm sure that will change as tool vendors improve things.

In the meantime, here are some valuable tools you can use to test for flaws in your Web 2.0 applications:

- **Firefox Web Developer** (`http://chrispederick.com/work/web-developer`) for analyzing script code and performing other manual checks.
- **SWFScan** (`https://h30406.www3.hp.com/campaigns/2009/wwcampaign/1-5TUVE/index.php?key=swf`) for analyzing Shockwave Flash (`.swf`) files.
- **WSDigger** (`www.foundstone.com/us/resources/proddesc/wsdigger.htm`) for analyzing Web services.
- **WSFuzzer** (`www.owasp.org/index.php/Category:OWASP_WSFuzzer_Project`) for analyzing Web services.

Web 2.0 applications are here to stay, so try to get your arms around their security issues now, before the technology grows even more complex.

Best Practices for Minimizing Web Security Risks

Keeping your Web applications secure requires ongoing vigilance in your ethical hacking efforts and on the part of your Web developers and vendors. Keep up with the latest hacks, testing tools, and techniques and let your developers and vendors know that security needs to be a top priority for your organization. I discuss getting security buy-in in Chapter 19.

You can gain direct hands-on experience testing and hacking Web applications by using the OWASP WebGoat Project (`www.owasp.org/index.php/Category:OWASP_WebGoat_Project`) and Foundstone's Hacme Tools `www.foundstone.com/us/resources-free-tools.asp`. I highly recommended you check them out and get your hands dirty!

Obscurity

The following forms of *security by obscurity* — hiding something from obvious view using trivial methods — can help prevent automated attacks from worms or scripts that are hard-coded to attack specific script types or default HTTP ports:

- To protect Web applications and related databases, use different machines to run each Web server, application, and database server.

  The operating systems on these individual machines should be tested for security vulnerabilities and hardened based on best practices and the countermeasures described in Chapters 10 through 12.

- Use built-in Web server security features to handle access controls and process isolation, such as the application-isolation feature in IIS.

  This helps ensure that if one Web application is attacked, it won't necessarily put any other applications running on the same server at risk.

- Use a tool for obscuring your Web server's identity — essentially anonymizing your server. For example, Port 80 Software's ServerMask (`www.port80software.com/products/servermask`).

- If you're concerned about platform-specific attacks being carried out against your Web application, you can trick the attacker into thinking the Web server or operating system is something completely different. Here are a few examples:

  - If you're running a Microsoft IIS server and applications, you might rename all your ASP scripts to have a `.cgi` extension.
  - If you're running a Linux Web server, use a program such as IP Personality (`http://ippersonality.sourceforge.net`) to change the OS fingerprint so the system looks like it's running something else.

- Change your Web application to run on a nonstandard port. Change from the default HTTP port 80 or HTTPS port 443 to a high port number, such as 8877, and, if possible, set the server to run as an unprivileged user — that is, something other than system, administrator, root, and so on.

Never *ever* rely on obscurity alone; it isn't foolproof. A dedicated attacker might determine that the system isn't what it claims to be.

Firewalls

Consider using additional controls to protect your Web systems, including:

- **A network-based firewall that can detect and block attacks against Web applications.** This includes commercial firewalls available from such companies as Juniper Networks — formerly NetScreen — (`www.juniper.net/us/en/products-services/security`), SonicWall (`www.sonicwall.com`), and Check Point (`www.checkpoint.com`).
- **A host-based Web application IPS,** such as SecurelIS (`www.eeye.com/html/products/secureiis/index.html`) or ServerDefender (`www.port80software.com/products/serverdefender`).

  These programs can detect Web application and certain database attacks in real time and cut them off before they have a chance to do any harm.

Source code analysis

Software development is where security holes begin and *should* end — but rarely do. If you feel confident in your ethical hacking efforts to this point, you can dig deeper to find security flaws in your source code — things that might never be discovered by traditional scanners and hacking techniques but that are problems nonetheless. Fear not — it's actually much simpler than it sounds. No, you won't have to go through the code line-by-line to see what's happening. You don't even need development experience (although it does help).

To do this, you can use a static source code analysis tool, such as those offered by Ounce Labs (`www.ouncelabs.com`) and Klockwork (`www.klocwork.com`). My favorite, CxDeveloper by Checkmarx (`www.checkmarx.com`), is available in North America through Security Innovation (`www.securityinnovation.com`).

With CxDeveloper, you simply tell it where the source code is located, as shown in Figure 14-13, pick the scan policy you wish to run, and then click Scan, and you're off and running.

When the scan completes, you can review the findings and recommended solutions, as shown in Figure 14-14.

CxDeveloper is pretty much all you need to analyze and report on vulnerabilities in your C, C++, C#, and Java source code bundled into one simple package. If you need to do more in-depth analysis performing customized queries, CxDeveloper's sister product, CxAudit, might be a good fit.

The bottom line with Web security is that if you can show your developers and quality assurance analysts that security begins with them, you can really make a difference in your organization's overall information security.

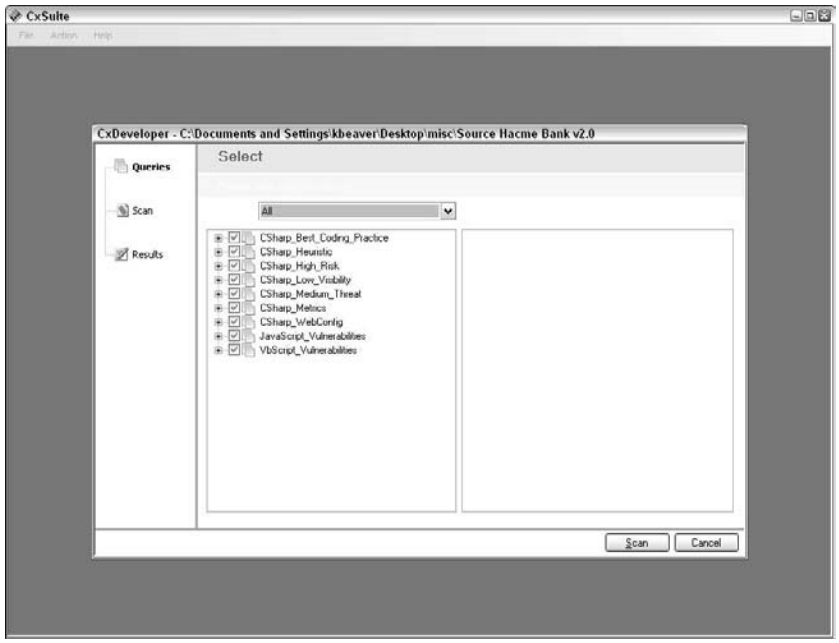


Figure 14-13: Using CxDeveloper to do an in-depth analysis of ASP.NET source code.

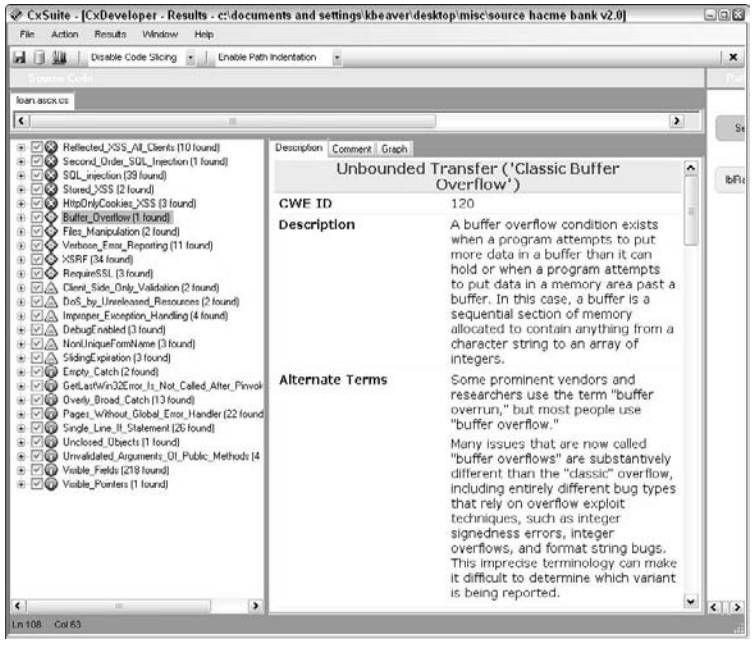


Figure 14-14: Reviewing the results of a CxDeveloper source code analysis.

Chapter 15

Databases and Storage Systems

In This Chapter

- Testing and exploiting database flaws
- Finding storage weaknesses
- Ferreting out sensitive information
- Countering database and storage abuse

Attacks against databases and storage systems can be very serious because that's where "the goods" are located — as the bad guys are well aware. These attacks can occur across the Internet or on the internal network when external attackers and malicious insiders exploit any number of vulnerabilities. These attacks can also occur via the Web application through SQL injection.

Databases

Database systems, such as Microsoft SQL Server, MySQL, and Oracle, have lurked behind the scenes but their value — and their vulnerabilities — have finally come to the forefront. Yes, even the mighty Oracle that was once claimed to be unhackable is susceptible to similar exploits as its competition. With the slew of regulatory requirements governing database security, hardly any business can hide from the risks that lie within because practically every business (large and small) uses some sort of database.

Choosing tools

As with wireless, operating systems, and so on, you need good tools if you're going to find the database security issues that count. My favorite tools for testing database security are

- **Advanced SQL Password Recovery** (`www.elcomsoft.com/asqlpr.html`) for cracking Microsoft SQL Server passwords

- **Cain & Abel** (`www.oxid.it/cain.html`) for cracking database password hashes
- **QualysGuard** (`www.qualys.com`) for performing in-depth vulnerability scans
- **SQLPing3** (`www.sqlsecurity.com/Tools/FreeTools/tabid/65/Default.aspx`) for locating Microsoft SQL Servers on the network, checking for blank SA passwords, and performing dictionary password-cracking attacks

You can also use exploit tools, such as Metasploit, for your database testing.

Finding databases on the network

The first step in discovering database vulnerabilities is to figure out where they're located on your network. Sounds funny, but many network admins I've met aren't even aware of various databases running in their environments. This is especially true for the free SQL Server Express database software that anyone can download and run on a workstation or test system.

I can't tell you how often I find sensitive production data, such as credit cards and Social Security numbers, being used in test databases that are completely wide open to abuse by curious insiders. Using sensitive data in the uncontrolled areas of development and quality assurance (QA) is a data breach waiting to happen.

The best tool I've found to discover Microsoft SQL Server systems is SQLPing3, shown in Figure 15-1.

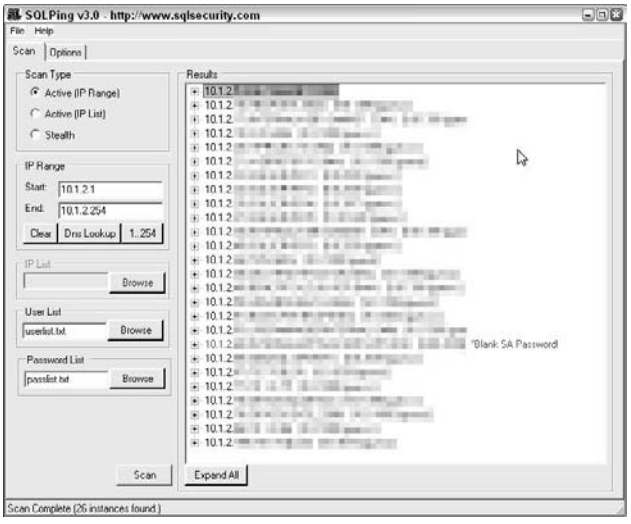


Figure 15-1: SQLPing3 can find SQL Server systems and check for missing SA account passwords.

A case study in hacking databases with Chip Andrews

The Situation

During a routine penetration test, Mr. Andrews performed the obligatory Google searches, domain name research, operating system fingerprinting, and port scans, but this particular Web site was locked down tight. Moving on to the Web-based application running on the system, he was immediately confronted with a login page using SSL-encrypted forms authentication. By checking the source of the Web page, he noticed that a hidden `App_Name` field was being passed to the application whenever a user attempted to log in to the site. Could it be that the developers might have failed to perform proper input validation on this innocent-looking parameter? The hunt was on . . . .

The Outcome

First, it was time to assemble the toolkit. At the time of this penetration test, Mr. Andrews preferred to use the following: Paros Proxy, Absinthe, Cain & Abel, Data Thief, and the Microsoft SQL Server Management Studio/SQL Server (Express Edition) — all of which are available free. For starters, he used Paros Proxy to allow for more control and visibility to the Web requests made to the Web server. After spidering the site for available pages and performing a quick vulnerability check for SQL injection, it was confirmed that the `App_Name` parameter appeared to cause the application to throw an Error 500 exception — indicating an application failure. Penetration tests are one of the rare occasions when an application failure is a desirable outcome.

Because the application failure indicated that Mr. Andrews could inject unintended characters into the SQL code being sent from the application to the database, he could see whether it was an exploitable condition. A common test that works with Microsoft SQL Server databases is to inject a command, such as `WAITFOR DELAY '00:00:10'`, which causes the database server to stall for 10 seconds. In an application that normally returns a page in one second or less, a consistent 10-second delay is a good indicator that you can inject commands into the SQL stream.

Next, Mr. Andrews attempted to use the Data Thief tool to attack the login page. This tool attempts to force the database to use an `OPENROWSET` command to copy data from the target database to Mr. Andrews's database located on the Internet. This is usually a very efficient way to siphon large amounts of data from vulnerable databases, but in this case, his attack was foiled! The database administrator at the target had disabled the `OPENROWSET` functionality by properly configuring the Disable Adhoc Distributed Queries option.

With diligence as his watchword, Mr. Andrews persisted with the next tool — Absinthe. This tool uses a technique called "blind SQL injection" to make determinations about data using simple yes or no questions of the database. For example, the tool might ask the database if the first letter of a table is less than "L." If yes, then the application might do nothing, but if no, the application might throw an exception. Using this simple binary logic, it is possible to use this technique to reveal the entire database structure and even the data stored inside — albeit very slowly. Using the tool, he identified a table of sensitive customer information and downloaded several hundred records to show the client.

(continued)

(continued)

Finally, it was time to attempt one last act of database dastardliness. First, Mr. Andrews loaded the tool called Cain & Abel and set it to enter sniffing mode. Then, using Paros Proxy and the already identified vulnerable parameter, he used the `xp_dirtree` extended stored procedure, which is available to all SQL Server database users, to attempt to show a directory on his Internet-connected machine using a Universal Naming Convention (UNC) path. This forced the target database to actually attempt to authenticate itself against Mr. Andrews's machine. Because Cain & Abel was listening on the wire, it obtained the hash of the challenge used to authenticate the exposed file share. By passing this hash to the password cracker built in to Cain & Abel, Mr. Andrews would have the username and password of the account under which the vulnerable SQL Server was running in just a matter of time (assuming it was not a local system account).

Would this hacked account use the same password as the admin account of the Web application? Would this password be the same as the local administrator account on the host? Those were questions for another day. It was time to assemble all the collected data, prepare a report for the client, and put the tools away for another day.

Chip Andrews is a co-founder of security consulting firm Special Ops Security, Inc. and owner of SQLSecurity.com (`http://sqlsecurity.com`), which has multiple resources on Microsoft SQL Server security, including the SQLPing3 tool. A coauthor for several books on SQL Server security (*Hacking Exposed: Windows Server 2003* and *SQL Server Security,* both published by McGraw-Hill Osborne) and a Black Hat presenter, Mr. Andrews has been promoting SQL Server and application security since 1999.

SQLPing3 can now discover instances of SQL Server hidden behind personal firewalls and more — a feature formerly only available in SQLPing2's sister application SQLRecon.

If you have Oracle in your environment, Pete Finnigan has a great list of Oracle-centric security tools at `www.petefinnigan.com/tools.htm` that can perform functions similar to SQLPing3.

Cracking database passwords

SQLPing3 also serves as a nice dictionary-based SQL Server password-cracking program. As you can see in Figure 15-1, it also checks for blank SA passwords by default. Another free tool for cracking SQL Server, MySQL, and Oracle password hashes is Cain & Abel, shown in Figure 15-2.

The commercial product Elcomsoft Distributed Password Recovery (`www.elcomsoft.com/edpr.html`) can also crack Oracle password hashes.

If you have access to SQL Server `master.mdf` files, you can use Elcomsoft's Advanced SQL Password Recovery (`www.elcomsoft.com/asqlpr.html`) to recover database passwords immediately.

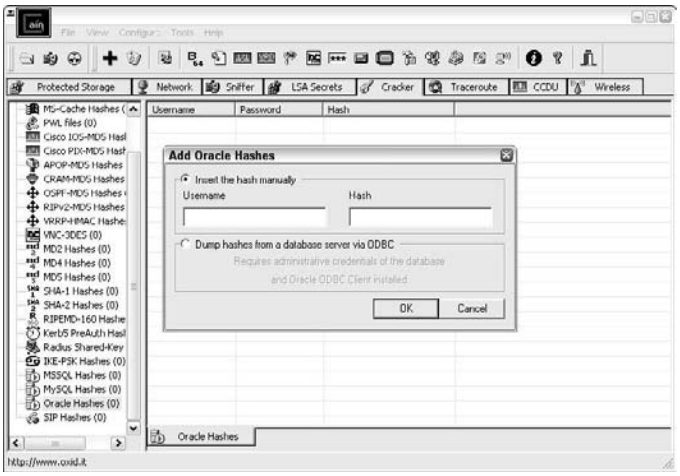


Figure 15-2: Using Cain & Abel to crack Oracle password hashes.

You might stumble across some legacy Microsoft Access database files that are password protected as well. No worries, the tool Advanced Access Password Recovery (`www.elcomsoft.com/acpr.html`) can get you right in.

As you can imagine, these password-cracking tools are a great way to demonstrate the most basic of weaknesses in your database security. One of the best ways to go about proving that there's a problem is to utilize Microsoft SQL Server Management Studio Express (`www.microsoft.com/express/sql/default.aspx`) to connect to the database systems you now have the passwords for and setup backdoor accounts or browse around to see what's available. In practically every unprotected SQL Server system I come across, there's sensitive personal financial or healthcare information available for the taking.

Scanning databases for vulnerabilities

As with operating systems and Web applications, some database-specific vulnerabilities can be rooted out only by using the right tools. I use QualysGuard to find such issues as

- Buffer overflows
- Privilege escalations
- Password hashes accessible through default/unprotected accounts
- Weak authentication methods enabled
- Database listener log files that can be renamed without authentication

Several all-in-one commercial database vulnerability scanners perform in-depth database checks on SQL Server, Oracle, and so on, such as NGSSQuirreL (`www.ngssoftware.com/products/database-security`) and AppDetectivePro (`www.appsecinc.com/products/appdetective`). They can be a good addition to your security testing tool arsenal if you can justify the investment.

Many vulnerabilities can be tested from both an unauthenticated outsider's perspective as well as a trusted insider's perspective. For example, you can use the SYSTEM account for Oracle to log in, enumerate, and scan the system (something that QualysGuard supports). My fingers are crossed that Qualys will eventually support authenticated scans for SQL Server.

Best Practices for Minimizing Database Security Risks

Keeping your databases secure is actually pretty simple if you do the following:

- Run your databases on different machines.
- Check the underlying operating systems for security vulnerabilities. I cover operating system exploits for Windows, Linux, and Novell NetWare in Chapters 10–12.
- Ensure that your databases fall within the scope of patching and system hardening.
- Require strong passwords on every database system.
- Use appropriate file and share permissions to keep prying eyes away.
- De-indentify any sensitive production data before it's used in development or QA.
- Check your Web applications for SQL injection and related input validation vulnerabilities.
- Use a network firewall, such as those available from Juniper Networks — formerly NetScreen — (`www.juniper.net/us/en/products-services/security`) or SonicWall (`www.sonicwall.com`), and database firewalls, such as those available from Imperva (`www.imperva.com/products/database-firewall.html`) and Pyn Logic (`www.pynlogic.com/enzoinfo2.aspx`).

- Run the latest version of database server software — especially if you're a Microsoft shop. The new security features in SQL Server 2008 and SQL Server Express are great advancements toward better database security.

Storage Systems

Attackers are carrying out a growing number of storage-related hacks. Hackers use various attack vectors and tools to break into the storage environment (surely, you know what I'm going to say next). Therefore, you need to get to know the techniques and tools yourself and use them to test your own storage environment.

There are a lot of misconceptions and myths related to the security of such storage systems as Fibre Channel and iSCSI Storage Area Networks (SANs), CIFS and NFS-based Network Attached Storage (NAS) systems, and so on. Many network and storage administrators believe that "Encryption or RAID equals storage security," "An external attacker can't reach our storage environment," or "Security is handled elsewhere." These are all very dangerous beliefs, and I'm confident that more attacks will target critical storage systems.

As with databases, practically every business has some sort of network storage housing sensitive information that it can't afford to lose. That's why it's very important to include both network storage (SAN and NAS systems) and traditional file shares into the scope of your ethical hacking.

Choosing tools

My favorite tools for testing storage security are

- **FileLocator Pro** (`www.mythicsoft.com/filelocatorpro`) for seeking sensitive information in unstructured files
- **Identity Finder** (`www.identityfinder.com`) for seeking sensitive information in unstructured files
- **LANguard** (`www.gfi.com/lannetscan`) for finding open and unprotected shares
- **QualysGuard** (`www.qualys.com`) for performing in-depth vulnerability scans
- **SuperScan** (`www.foundstone.com/us/resources/proddesc/superscan.htm`) for port scanning to find live storage hosts

You should also have this group of niche storage security testing tools on your radar:

- **CHAP Password Tester** (`www.isecpartners.com/cpt_chap_password_tester.html`)
- **CIFShareBF** (`www.isecpartners.com/SecuringStorage/CIFShareBF.zip`)
- **GrabiQNs** (`www.isecpartners.com/SecuringStorage/GrabiQNs.zip`)
- **NASanon** (`www.isecpartners.com/SecuringStorage/NASanon.zip`)
- **StorScan** (`www.isecpartners.com/storscan.html`)

Finding storage systems on the network

To seek out storage-related vulnerabilities, you have to figure out what information is where. The best way to get rolling is to use a port scanner and, ideally, an all-in-one vulnerability scanner, such as QualysGuard or LANguard. Also, given that many storage servers have Web servers built in, you can use such tools as Acunetix Web Vulnerability Scanner and WebInspect to uncover Web-based flaws. You can use these vulnerability scanners to gain good insight into areas that need further inspection, such as weak authentication, DNS server name pollution, unpatched operating systems, unprotected Web servers, and so on.

A commonly overlooked storage vulnerability is that many storage systems can be accessed from both the DMZ and the internal network. This vulnerability poses risks to both sides of the network. Be sure to manually assess whether you can reach the DMZ from the internal network and vice versa.

You can also perform basic file permission and share scans (as outlined in Chapters 10 and 11) in conjunction with a text search tool to uncover sensitive information that everyone on the network should not have access to.

Rooting out sensitive text in network files

An important "authenticated" test to run on your storage systems is to scan for sensitive information stored in readily accessible text files. It's as simple as using a text search utility, such as FileLocator Pro or Effective File Search (`www.sowsoft.com/search.htm`). You can even use Google Desktop

(`http://desktop.google.com`) if you prefer. Alternatively, you can use Windows Explorer to scan for sensitive information, but it's just too slow and cumbersome for my liking.

You'll be *amazed* at what you come across stored unsecurely on users' Windows desktops, server shares, and more, such as

- Employee health records
- Customer credit card numbers
- Corporate financial reports

Such sensitive information should not only be protected by good business practices, but they're also governed by state, federal, and international regulations.

Do your searches for sensitive text while you're logged in to the local system or domain as a regular user — not an administrator. This will give you a better view of regular users who have unauthorized access to sensitive files and shares that you thought were otherwise secure. When using a basic text search tool, such as FileLocator Pro, look for the following text strings:

- DOB (for dates of birth)
- SSN (for Social Security numbers)
- License (for driver's license information)
- Credit (for credit card numbers)

Don't forget about your mobile devices when seeking sensitive, unprotected information. Everything from laptops to USB drives to external hard drives is fair game to the bad guys. A costly data breach is as close as a misplaced or stolen system.

The possibilities for information exposure are endless; just start with the basics and only peek into nonbinary files that you know will have text in them. Limiting your search to these text-based files will save you a ton of time!

- `.txt`
- `.doc` and `.docx`
- `.dbf`
- `.db`
- `.rtf`
- `.xls` and `.xlsx`

An example of a basic text search using FileLocator Pro is shown in Figure 15-3. Note the files found in different locations on the server.

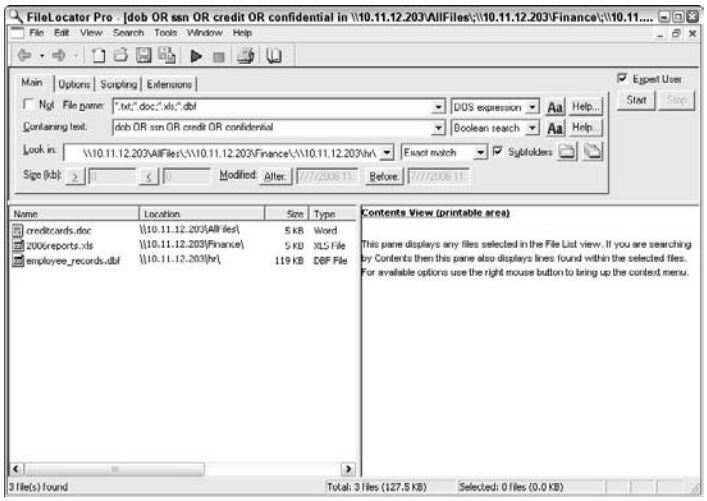


Figure 15-3: Using FileLocator Pro to search for sensitive text on unprotected shares.

To speed the process, you can use Identity Finder, a tool designed for the very purpose of scanning storage devices for sensitive, personally identifiable information. Figure 15-4 shows what such a tool can find in just a matter of minutes.

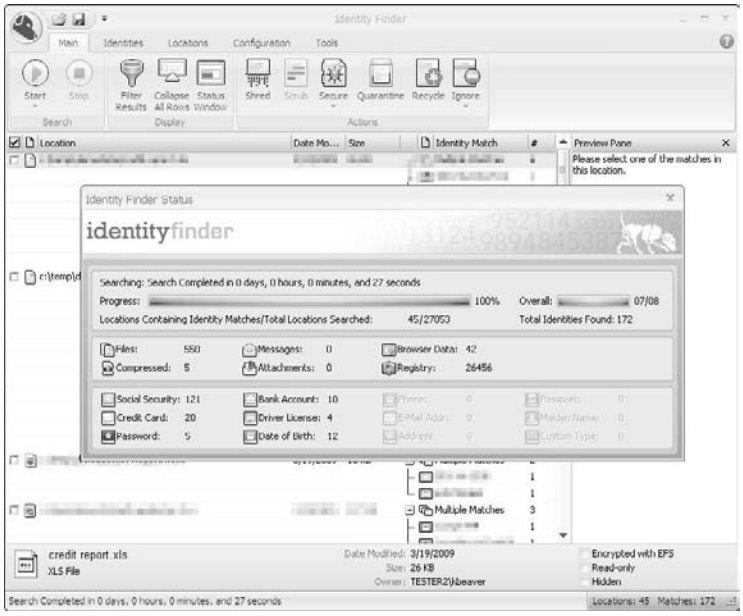


Figure 15-4: Using Identity Finder to uncover hundreds of sensitive records on an unprotected storage device.

Identity Finder has an Enterprise edition that can be used to search network systems and even databases for sensitive information.

For a second round of testing, you could perform your searches logged in as administrator. You're likely to find a lot of sensitive information scattered about. It seems worthless at first; however, this can highlight sensitive information stored in places it shouldn't be or that the network administrator shouldn't have access to.

Testing is highly dependent on timing, searching for the right keywords, and looking at the right systems on the network. You likely won't root out every single bit of sensitive information, but this will show you where certain problems are to help justify the need for stronger access controls and better IT and security management processes.

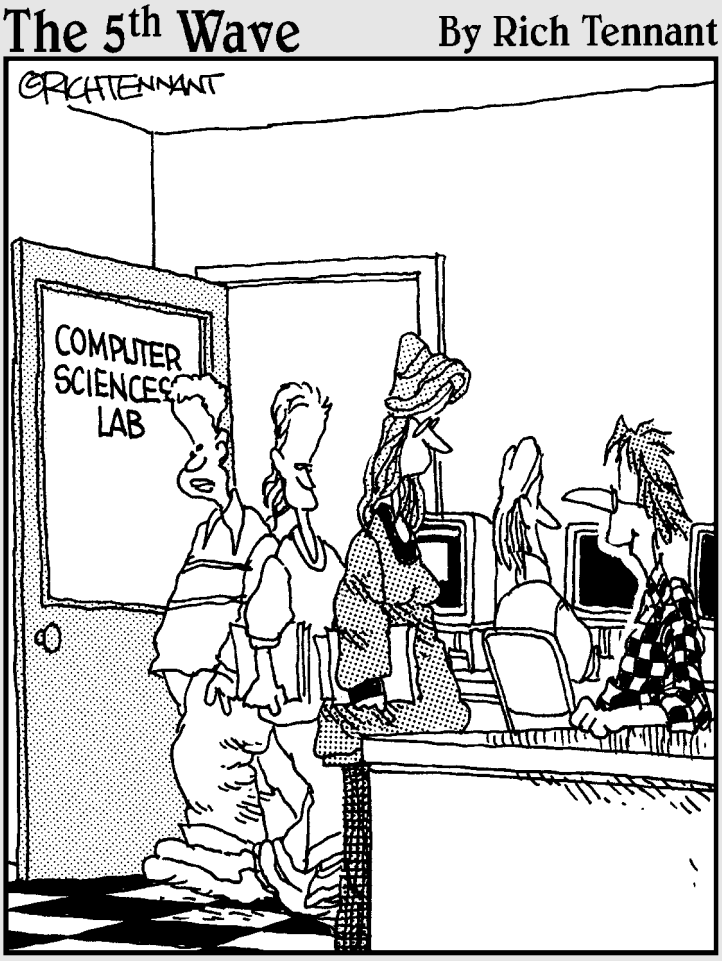
Best Practices for Minimizing Storage Security Risks

Like database security, storage security is not rocket surgery. Keeping your storage systems secure is also simple if you do the following:

- Check the underlying operating systems for security vulnerabilities. I cover operating system exploits for Windows, Linux, and Novell NetWare in Chapters 10–12.
- Ensure that your network storage (SAN and NAS systems) falls within the scope of patching and system hardening.
- Require strong passwords on every storage management interface.
- Use appropriate file and share permissions to keep prying eyes away.
- Educate your users on where to store sensitive information and the risks of mishandling it.
- De-indentify any sensitive production data before it's used in development or QA.
- Use a network firewall, such as those available from Juniper Networks — formerly NetScreen — (`www.juniper.net/us/en/products-services/security`) and SonicWall (`www.sonicwall.com`) to properly segment your internal network.

Part VI

Ethical Hacking Aftermath



"I'm sure there will be a good job market when I graduate. I created a virus that will go off that year."

In this part . . .

Now that the hard — or at least technical — stuff is over with, it's time to pull everything together, fix what's broken, and establish some good information security practices to move forward with.

First, this part covers reporting the security vulnerabilities you discover to help get management buy-in and hopefully more budget to make things right. This part then covers some good practices for plugging the various security holes within your systems and patching everything up to keep from being attacked. Finally, this part covers what it takes to manage change within your security systems for long-term success, including outsourcing ethical hacking so you can add even more projects to your overflowing plate! That's what working in IT and compliance is all about anyway, right?

Chapter 16

Reporting Your Results

In This Chapter

- Bringing your test data together
- Categorizing vulnerabilities you discover
- Documenting and presenting the results

If you're looking for a break after testing, now isn't the time to rest on your laurels. The reporting phase of your ethical hacking is one of the most critical pieces. The last thing you want to do is to run your tests, find security problems, and leave it at that. Put your time and effort to good use by thoroughly analyzing and documenting what you find to ensure that security vulnerabilities are eliminated and your information is more secure as a result. This is an essential element of the ongoing vigilance that information security and risk management requires.

Ethical hacking reporting includes sifting through all your findings to determine which vulnerabilities need to be addressed and which ones don't really matter. Reporting also includes briefing management or your client on the various security issues you find, as well as giving specific recommendations for making improvements. You share the information you've gathered and give the other parties guidance on where to go from there. Reporting also shows that the time, effort, and money invested in the ethical hacking tests were put to good use.

Pulling the Results Together

When you have gobs of test data — from screenshots and manual observations you documented to detailed reports generated by the various vulnerability scanners you used — what do you do with it all? You need to go through your documentation with a fine-toothed comb and highlight all the areas that stand out. Base your decisions on

- Vulnerability rankings from your assessment tools
- Your knowledge as a security professional
- The context of the vulnerability

So that you can find out more information about the vulnerability, many feature-rich security tools assign each vulnerability a ranking (based on overall risk), explain the vulnerability, give possible solutions, and reference links to vendor sites, the Common Vulnerabilities and Exposures Web site at `http://cve.mitre.org`, and the National Vulnerabilities Database at `http://nvd.nist.gov`. For further research, you might also need to reference other support sites and online forums to see whether the vulnerability affects your particular system and situation. Overall business risk is your main focus.

You can plug in this information to a table in Excel or in Word. I prefer to go through everything in hard-copy form because it's easier for me to read, but your choice might depend on how much data you have. If you think more highly of trees, you might want to just read the results on the computer screen and copy and paste the items that stand out into your final report.

In your final report document, you might want to organize the vulnerabilities as shown in the following list:

- Nontechnical issues
  - Social engineering vulnerabilities
  - Physical security vulnerabilities
  - IT operations weaknesses
  - Other
- Workstations and servers
  - Operating systems
  - Other
- Applications
- Publicly accessible
- Internal
- Database systems
- Network infrastructure systems
  - Hubs and switches
  - Routers
  - Firewalls
  - Intrusion detection systems
  - Wireless access points
  - Other

For further clarity, create separate lists for these categories of security vulnerabilities:

- Internal vulnerabilities, such as internal hosts and operational issues
- External vulnerabilities, such as public hosts, business partner network connections, and telecommuters

The formatting of your report ultimately comes down to your own style and the feedback you get from the other people who read the report. There's no right or wrong here.

Prioritizing Vulnerabilities

Prioritizing the security vulnerabilities you find is critical because many issues might not be fixable and others might not be worth fixing. You might not be able to eliminate some vulnerabilities because of various technical reasons, and you might not be able to afford to eliminate others. You need to factor whether the benefit is worth the effort and cost. For instance, if you determine that it will cost $30,000 to encrypt a sales leads database worth $20,000 to the organization, encryption might not make sense. On the other hand, spending a few weeks worth of development time to fix cross-site scripting and SQL injection vulnerabilities could be worth a lot of money. You need to study each vulnerability carefully, determine the business risk, and weigh whether the issue is worth fixing.

Analyze each vulnerability carefully and determine your worst-case scenarios. It's impossible — or at least not worth trying — to fix every vulnerability that you find.

Here's a quick method to use when prioritizing your vulnerabilities that you can tweak to accommodate your needs. You need to consider two major factors for each of the vulnerabilities you discover:

- **Likelihood of exploitation:** How likely is it that the specific vulnerability you're analyzing will be taken advantage of by a hacker, a malicious user, malware, or other threat?
- **Impact if exploited:** How detrimental would it be if the vulnerability you're analyzing were exploited?

Many people often skip these considerations and assume that every vulnerability discovered has to be resolved. Big mistake. Just because a vulnerability is discovered doesn't mean it applies to your particular situation and environment. If you go in with the mindset that every vulnerability will be addressed regardless of circumstances, you'll waste a lot of unnecessary

time, effort, and money and can set up your ethical hacking program for failure in the long term. However, be careful not to swing too far in the other direction! Many vulnerabilities don't appear too serious on the surface but could very well get your organization into hot water if they're exploited.

Rank each vulnerability, using criteria such as High, Medium, and Low or a 1-through-5 rating (where 1 is the lowest priority and 5 is the highest) for each of the two considerations. Table 16-1 shows a sample table and a representative vulnerability for each category.

Table 16-1 Prioritizing Vulnerabilities

	High Likelihood	*Medium Likelihood*	*Low Likelihood*
High Impact	Sensitive information stored on an unencrypted laptop	Tape backups taken offsite that are not encrypted and/or password protected	No admin password on a SQL Server system
Medium Impact	Unencrypted emails being sent	Missing Windows patch on internal server that can be exploited using Metasploit	No passwords required on several Windows administrator accounts
Low Impact	Outdated virus signatures on a standalone PC dedicated to Internet browsing	Cleaning crew personnel gaining unauthorized network access	Weak SSL encryption being exploited on e-commerce site

The vulnerability prioritization shown in Table 16-1 is based on the qualitative method of assessing security risks. It's subjective, based on your knowledge of the systems and vulnerabilities, but you can also consider any risk ratings you get from your security tools — just don't rely solely on them because a vendor can't provide ultimate rankings of vulnerabilities. If you need to go more in-depth on your risk analysis, you should check out the OCTAVE methodology developed and published by the CERT Coordination Center's Software Engineering Institute (`www.cert.org/octave`).

Reporting Methods

You may need to organize your vulnerability information into a formal document for management or your client. This is not always the case, but it's often the professional thing to do and shows that you take your work seriously. Ferret out the critical findings and document them so that other parties can understand them.

Graphs and charts are a plus. Screen captures of your findings — especially when it's difficult to save the data to a file — can add a nice touch to your reports and show tangible exploitations.

Document the vulnerabilities in a concise, nontechnical manner. Every report should contain the following information:

- Dates and times the testing was carried out
- Tests that were performed
- Summary of the vulnerabilities discovered
- Prioritized list of vulnerabilities that need to be addressed

If it will add value to management or your client (and it often does), add this information to your report:

- Recommendations and specific steps on how to plug the security holes found
- List of general recommendations to improve overall security

Most people want the hard copy report to include a *summary* of the findings — not everything. The last thing most people want to do is sift through a 5-inch-thick stack of papers containing technical jargon that means very little to them.

Many managers and clients like receiving raw data reports from the security tools on a CD-ROM or an encrypted ZIP file via e-mail. That way, they can reference the data later if they want but aren't mired in hundreds of hard-copy pages of technical gobbledygook.

Your list of action items in your report might include

- Enable Windows auditing on all servers.
- Put a secure lock on the server-room door.
- Harden operating systems based on strong security practices from the National Vulnerabilities Database (`http://crsc.nist.gov`), the Center for Internet Security Benchmarks/Scoring Tools (`www.cisecurity.org`), and *Network Security For Dummies*.
- Harden your wireless access point by using the techniques and recommendations presented in *Hacking Wireless Networks For Dummies*.
- Use a cross-cut paper shredder for the destruction of confidential hard-copy information.
- Install personal firewall/IPS software on all laptops.

- Validate input in all Web applications to eliminate cross-site scripting and SQL injection.
- Apply the latest vendor patches to the database server.

As part of the final report, you might want to document employee reactions that you observe when carrying out your ethical hacking tests. For example, are employees completely oblivious or even belligerent when you carry out an obvious social engineering attack? Does the IT or security staff completely miss technical tip-offs, such as the performance of the network degrading during testing or various attacks appearing in system log files? You can also document other security issues you observe, such as how quickly IT staff or manager services providers respond to your tests or whether they respond at all.

Guard the final report to keep it secure from people who are not authorized to see it. An ethical hacking report and the associated documentation and files in the hands of a competitor, hacker, or malicious insider, could spell trouble for the organization. Here are some ways to prevent this from happening:

- Deliver the report and associated documentation and files only to those who have a business need to know.
- When e-mailing the final report, encrypt all attachments, such as documentation and test results, using PGP, encrypted ZIP format, and so on, and then share the password with the recipient via telephone or other secure communication method.
- Remove programs and data from the report that a hacker or insider could use in malicious ways, such as tools used (password crackers and network analyzers), log files, and test data.
- Leave the actual testing steps that a malicious person could abuse out of the report. Answer any questions on that subject as needed.

Chapter 17

Plugging Security Holes

In This Chapter

- Determining which vulnerabilities to address first
- Patching your systems
- Looking at security in a new light

After you complete your tests, you want to head down the road to greater security. However, you found some security vulnerabilities. (Hopefully not too many serious ones, though!) Plugging these security holes before a hacker exploits them is going to require a little elbow grease. You need to come up with your game plan and decide which security vulnerabilities to address first. A few patches might be in order, and possibly even some system hardening. You might want to reevaluate your network design and security infrastructure as well. I touch on some of the critical areas in this chapter. You might also want to refer to the fine book *Network Security For Dummies* by Chey Cobb. Chey does a great job of covering each of these topics in depth.

Turning Your Reports into Action

It might seem that the security vulnerability to address first would be obvious, but it's often not black and white. When reviewing the vulnerabilities that you find, consider the following variables:

- Whether the vulnerability can be fixed
- How easy the vulnerability is to fix
- How critical the vulnerable system is
- Whether you can take the system offline to fix the problem
- Time, money, and effort involved in purchasing new hardware or software or retooling business processes to plug the holes

In Chapter 16, I cover the basic issues of determining how important and how urgent the security problem is. In fact, I provide real-world examples in Table 16-1. You should also look at security from a time-management perspective and address the issues that are both important (high impact) and urgent (high likelihood). You don't want to try to fix the vulnerabilities that are *just* high impact or *just* high likelihood. You might have some high-impact vulnerabilities that, likely, are never exploited. Likewise, you probably have some vulnerabilities with a high likelihood of being exploited that, if they are exploited, won't really make a big difference in your business or your job. This type of human analysis and perspective will keep security professionals employed for some time to come!

Focus on tasks with the highest payoff first — those that are both high impact *and* high likelihood. Ideally, this will be the minority of your vulnerabilities. After you plug the most critical security holes, you can go after the less important and less urgent tasks when time and money permit. For example, after you plug such critical holes as SQL injection in Web applications and missing patches on important servers, you might want to reconfigure your tape backups with passwords, if not strong encryption, to keep prying eyes away in case your backups fall into the wrong hands.

Patching for Perfection

Do you ever feel like all you do is patch your systems to fix security vulnerabilities? If you answer yes to this question, good for you — at least you're doing it! If you constantly feel pressure to patch your systems the right way but can't seem to find time — at least it's on your radar. Many IT professionals and their managers don't even think about proactively patching their systems until after a breach occurs. If you're reading this book, you're obviously concerned about security and are hopefully way past that.

Whatever you do, whatever tool you choose, and whatever procedures work best in your environment — keep your systems patched! This goes for servers and workstations as well as operating systems and databases.

Patching is unavoidable. The only real solution to eliminating the need for patches is developing secure software in the first place, but that's not going to happen any time soon. A large portion of security incidents can be prevented with some good patching practices, so there's simply no reason not to have a solid patch management process in place.

Patch management

If you can't keep up with the deluge of security patches for all your systems, don't despair; you can still get a handle on the problem. Here are my basic tenets of applying patches to keep your systems secure:

- Make sure all the people and departments that are involved in applying patches on your organization's systems are on the same page and follow the same procedures.
- Have formal and documented procedures in place for these critical processes:
  - Obtaining patch alerts from your vendors
  - Assessing which patches affect your systems
  - Determining when to apply patches
- Make it policy and have a procedure in place for testing patches *before* you apply them to your production servers, if that's possible. Testing patches after you apply them isn't a big deal on workstations but servers are a different story. Many patches have "undocumented features" and subsequent unintended side effects — believe me, I've experienced this before. An untested patch is an invitation for system (and job) termination!

Patch automation

The following sections describe the various patch deployment tools you can use to lower the burden of constantly having to keep up with patches.

Commercial tools

I recommend a robust patch automation application — especially if you have

- A large network
- A network with several different operating systems (Windows, Linux, NetWare, and so on)
- More than a dozen computers

Be sure to check out these patch-automation solutions:

- BigFix (`www.bigfix.com`)
- Shavlik Technologies NetChk (`www.shavlik.com`)

- Ecora Patch Manager (`www.ecora.com/ecora/products/patchmanager.asp`)
- ScriptLogic Patch Authority Ultimate (`www.scriptlogic.com/products/patchauthorityultimate`)
- Windows Server Update Services from Microsoft (`www.microsoft.com/windowsserversystem/updateservices/default.mspx`)

The GFI LANguard (`www.gfi.com/lannetscan`) product that I use in this book can check for patches to apply and deploy.

Watch the other major vulnerability assessment tool vendors, such as Qualys. They are starting to integrate logic in their programs to deploy patches that address the vulnerabilities their products find — a process called *vulnerability management*.

Free tools

If you're running Windows, use one of these free tools to help with automated patching:

- Microsoft Update, which is built in to Microsoft Windows systems
- Microsoft Baseline Security Analyzer (MBSA), found at `www.microsoft.com/technet/security/tools/mbsahome.mspx`

Hardening Your Systems

After you patch your systems, you have to make sure your systems are hardened from the other security vulnerabilities that patches can't fix. I've found that many people stop with patching, thinking their systems are secure, but that's just not possible. Throughout the years, I've seen network administrators ignore recommended hardening practices from such organizations as the National Institute of Standards and Technology (NIST) (`http://csrc.nist.gov/publications/nistpubs/index.html`) and the Center for Internet Security (`www.cisecurity.org`), leaving many security holes wide open. However, I'm a true believer that hardening systems from malicious attack is not foolproof, either. Because every system and every organization's needs are different, there is no one-size-fits-all solution, so you have to strike a balance and not rely on any single option too much.

Chey Cobb's *Network Security For Dummies* contains many great resources for hardening various systems on your network.

This book presents hardening countermeasures that you can implement for your network, computers, and even physical systems and people. I find these countermeasures work the best for the respective systems.

Paying the piper

I was once involved in cleaning up a hack attack on a Windows NT server for a client. I had been saying ever since I was hired that I needed to harden the customer's network from attack. The customer had a Windows NT server wide open on the Internet with a public IP address (ouch!) *and* no firewall installed. The customer was willing to pay me to patch the server, but that was it. Therefore, I could do only so much to secure it from the elements, given their environment and specific needs. The customer didn't heed my advice on getting the server behind a firewall at minimum, if not reconfiguring the application so its security could be improved.

Time passed without incident, until one day, a hacker compromised the customer's Windows NT system, uploaded FTP server software, and then started hosting illegal movies and music — which almost immediately killed their Internet connection, effectively locked everyone (including customers) out of the server, and put a halt to the customer's e-commerce. After the downtime, lost business, and paying me to fix the problem, the customer spent a lot more than the price of a firewall and a couple of hours of configuration time that I asked them to invest up front.

Implementing at least the basic security practices is critical. Whether installing a firewall on the network or requiring users to have strong passwords — you *must* do the basics if you want any modicum of security. Beyond patching, if you follow the countermeasures I document, add the other well-known security practices for network systems (routers, servers, workstations, and so on) that are freely available on the Internet, and perform ongoing ethical hacking tests, you can rest assured that you're doing your best to keep your organization's information secure.

Assessing Your Security Infrastructure

A review of your overall security infrastructure can add oomph to your systems:

- **Look at how your network and building are designed.** Consider organizational issues, such as whether policies are in place, maintained, or even taken seriously. Does management have buy-in on information security and compliance, or do they simply shrug the measure off as an unnecessary expense or barrier to conducting business?
- **Map your network by using the information you gather from the ethical hacking tests in this book.** Updating existing documentation is a major necessity. Outline IP addresses, running services, and whatever else you discover. Draw your network diagram — network design and

overall security issues are a whole lot easier to assess when you can work with them visually. Although I prefer to use a technical drawing program, such as Visio, to create network diagrams, such a tool isn't necessary — you can sketch your map on a napkin!

Be sure to update your diagrams when your network changes.

- **Think about your approach to correcting vulnerabilities and increasing your organization's overall security.** Are you focusing all your efforts on the perimeter and not on a layered security approach? Think about how most convenience stores and banks are protected. Security cameras focus on the cash registers, teller computers, and surrounding areas — not just on the parking lot or entrances. Look at security from a *defense in-depth* perspective. Make sure that several layers of security are in place in case one measure fails, so the malicious attacker must go through other barriers to carry out a successful hack attack.
- **Think about security policies and procedures at an organizational level.** Document what security policies and procedures are in place and whether they're effective. Look at the overall security culture within your organization and see what it looks like from an outsider's perspective. What would customers or business partners think about how your organization treats their sensitive information?

Looking at your security from a high-level and nontechnical perspective gives you a new outlook on security holes. It takes some time and effort at first, but after you establish a baseline of security, it's much easier to manage new threats and vulnerabilities.

Chapter 18

Managing Security Changes

In This Chapter

- Automating tasks
- Watching for misbehavior
- Outsourcing testing
- Keeping security on everyone's mind

Information security is an ongoing process that you must manage effectively to be successful. This goes beyond periodically applying patches and hardening systems. Performing your ethical hacking tests repeatedly is critical; information security threats and vulnerabilities emerge constantly. Additionally, ethical hacking tests are just a snapshot of your overall information security, so you *have* to perform your tests continually to keep up with the latest security issues. Ongoing vigilance is not only required for compliance with various laws and regulations but also for minimizing business risks related to your information systems.

Automating the Ethical Hacking Process

You can run a large portion of the ethical hacking tests in this book automatically if you have the right tools:

- Ping sweeps and port scans to show what systems are available and what's running
- Password-cracking tests to attempt access to external Web applications, remote access servers, and so on
- Vulnerability scans to check for missing patches, misconfigurations, and exploitable holes
- Exploitation of vulnerabilities (to an extent, at least)

You must have the right tools to automate tests:

- Some commercial tools can set up ongoing assessments and create nice reports for you without any hands-on intervention — just a little setup and scheduling time up front. This is why I like many of the commercial — and mostly automated — security testing tools, such as QualysGuard and WebInspect. The automation you get from these tools often helps justify the price — especially because you don't have to be up at 2:00 a.m. or on call 24 hours a day to monitor the testing.
- Standalone security tools, such as Nmap, John the Ripper, and Netstumbler, aren't enough. You can use the Windows Scheduler and AT commands on Windows systems and cron jobs on UNIX-based systems, but manual steps and human intellect are still required.

You can't gain true security if you automate *everything.* Certain tests and phases, such as enumeration of new systems, various Web application tests, social engineering, and physical security walkthroughs, can't be set on autopilot — you have to be involved.

Even the smartest computer "expert system" can't accomplish some security tests. Good security requires both technical expertise and experience.

Monitoring Malicious Use

Monitoring security-related events is essential for ongoing security efforts. This can be as basic and mundane as monitoring log files on routers, firewalls, and critical servers every day, or as advanced and expensive as implementing a correlation security incident management system to monitor every little thing that's happening in your environment. A common method is to deploy an intrusion prevention system or data leakage prevention system and monitor for malicious behavior. The problem with monitoring security-related events is that humans find it very boring and very difficult to do effectively.

Each day consider dedicating a time — such as first thing in the morning — to check your critical log files from the previous night or weekend to ferret out intrusions and other computer and network security problems. You could dedicate a person to this task, but do you really want to subject someone to that kind of torture?

- Finding critical security events in system log files is difficult, if not impossible. It's just too tedious a task for the average human to accomplish effectively.

- Depending on the type of logging and security equipment you use, you might not even detect some security events, such as IDS evasion techniques and hacks coming into allowed ports on the network.

Enable system logging where it's reasonable and possible. You don't necessarily need to capture all computer and network events, but you should definitely look for certain obvious ones, such as login failures, malformed packets, and unauthorized file access. The preferable way to log security events is to use a syslog or other central server on your network. Do not keep logs on the local host, if possible, to help prevent the bad guys from tampering with log files to cover their tracks. Check out `www.loganalysis.org` for some good logging resources.

A couple of good solutions to the security-monitoring dilemma are

- **Purchase an event logging system.** A few low-priced yet effective solutions are available, such as GFI's EventsManager (`www.gfi.com/eventsmanager`). Typically, lower-priced event logging systems usually support only one OS platform — Microsoft Windows is the most common. Higher-end solutions, such as ArcSight's Logger (`www.arcsight.com/products/products-logger`), offer both log management across various platforms and event correlation to help track down the source of security problems and the various systems affected during an incident.
- **Outsource security monitoring to a third-party managed security services provider (MSSP).** Dozens of MSSPs were around during the Internet boom, but only a few strong ones remain, such as BT's Counterpane managed service (`http://bt.counterpane.com/index.html`) and SecureWorks (`www.secureworks.com`). The value in outsourcing security monitoring is that the MSSP often has facilities and tools that you would likely not be able to afford and maintain. They also have analysts working around the clock and have the security experiences and knowledge they gain from other customers to share with you.

  When MSSPs discover a security vulnerability or intrusion, they can usually address the issue immediately, often without your involvement. I recommend at least checking whether third-party firms and their services can free some of your time and resources so that you can focus on other things. Just don't depend solely on their monitoring efforts; an MSSP will have trouble catching insider abuse, social engineering attacks, and Web application hacks over SSL. You still need to be involved.
- **Outsource security scanning to a third-party, Software as a Service (SaaS) provider.** The latest trend is for organizations to outsource their external security scans to a third-party provider. As with MSSPs, organizations can often get up and running with little to no investment, and have the benefit of saying an independent third party is performing the scans.

Outsourcing Ethical Hacking

Outsourcing ethical hacking is very popular and a great way for organizations to get an unbiased third-party perspective of their information security. Outsourcing allows you to have a checks-and-balances system that clients, business partners, and regulators like to see.

Outsourcing ethical hacking can be expensive. Many organizations spend thousands of dollars — often tens of thousands — depending on the testing needed. However, doing all this yourself isn't cheap — and quite possibly not as effective, either!

A lot of confidential information is at stake, so you must trust your outside consultants and vendors. Consider the following questions when looking for an independent expert or vendor to partner with:

- **Is your ethical hacking provider on your side or a third-party vendor's side? Is the provider trying to sell you products, or is the provider vendor neutral?** Many providers might try to make a few more dollars off the deal — which might not be necessary for your needs. Just make sure that these potential conflicts of interest aren't bad for your budget and your business.
- **What other IT or security services does the provider offer? Does the provider focus solely on security?** Having an information security specialist do this testing for you is often better than working with an IT generalist organization. After all, would you hire a general corporate lawyer to help you with a patent, a general family practitioner to perform surgery, or a computer technician to rewire your house?
- **What are your provider's hiring and termination policies?** Look for measures the provider takes to minimize the chances that an employee will walk off with your sensitive information.
- **Does the provider understand your business needs?** Have the provider repeat the list of your needs and put them in writing to make sure you're both on the same page.
- **How well does the provider communicate?** Do you trust the provider to keep you informed and follow up with you in a timely manner?
- **Do you know exactly who will perform the tests?** Will one person do the testing, or will subject-matter experts focus on the different areas? (This isn't a deal breaker but is nice to know.)
- **Does the provider have the experience to recommend practical and effective countermeasures to the vulnerabilities found?** The provider shouldn't just hand you a think report and say, "Good luck with all that!" You need realistic solutions.

Thinking about hiring a *reformed* hacker?

Former hackers — I'm referring to the black-hat hackers who have hacked into computer systems in the past — can be very good at what they do. Many people swear by hiring reformed hackers to do ethical hacking. Others compare this to hiring the proverbial fox to guard the hen house. If you're thinking about bringing in a former unethical hacker to test your systems, consider these issues:

- Do you really want to reward malicious behavior with your organization's business?
- Claiming to be reformed doesn't mean he or she is. There could be deep-rooted psychological issues or character flaws you're going to have to contend with. *Buyer beware!*
- Information gathered and accessed during ethical hacking is some of the most sensitive information your organization possesses. If this information gets into the wrong hands — even ten years down the road — it could be used against your organization. Some hackers and reformed criminals hang out in tight social groups. You might not want your information shared in their circles.

That said, everyone deserves a chance to explain what happened in the past. Zero tolerance is senseless. Listen to his or her story and use common-sense discretion as to whether you trust this person to help you. The supposed black-hat hacker actually might have been a gray-hat hacker or a misguided white-hat hacker who fits well in your organization.

- **What are the provider's motives?** Do you get the impression that the provider is in business to make a quick buck off the services, with minimal effort and value added, or is the provider in business to build loyalty with you and establish a long-term relationship?

Finding a good organization to work with long term will make your ongoing efforts much simpler. Ask for several references and sample *sanitized* deliverables from potential providers. If the organization can't produce these without difficulty, look for another provider.

Your provider should have its own service agreement for you that includes a mutual nondisclosure statement. Make sure you both sign this to help protect your organization.

Instilling a Security-Aware Mindset

Your network users are often your first and last line of defense. Make sure your ethical hacking efforts and the money spent on your information security initiatives aren't wasted because a simple employee slip-up gave a malicious attacker the keys to the kingdom.

These elements can help establish a security-aware culture in your organization:

- **Make security awareness and training an active and ongoing process among all employees and users on your network, including management and contractors.**
- **Treat awareness and training programs as a long-term business investment.**

  Security awareness programs don't have to be expensive. You can buy posters, mouse pads, screen savers, pens, and sticky notes to help keep security on everyone's mind. Some creative solutions vendors are Greenidea, Inc. (`www.greenidea.com`), Security Awareness, Inc. (`www.securityawareness.com`), and The Security Awareness Company (`www.thesecurityawarenesscompany.com`).
- **Get the word on security out to management!**
- **Align your security message with your audience and keep it as nontechnical as possible.**
- **Lead by example.** Show that you take security seriously and offer evidence that helps prove that everyone else should, too.

If you can get the ear of management *and* users, and put forth enough effort to make security a priority day after day, you can help shape your organization's culture. This can provide security value beyond your wildest imagination. I've seen the difference it makes!

Keeping Up with Other Security Issues

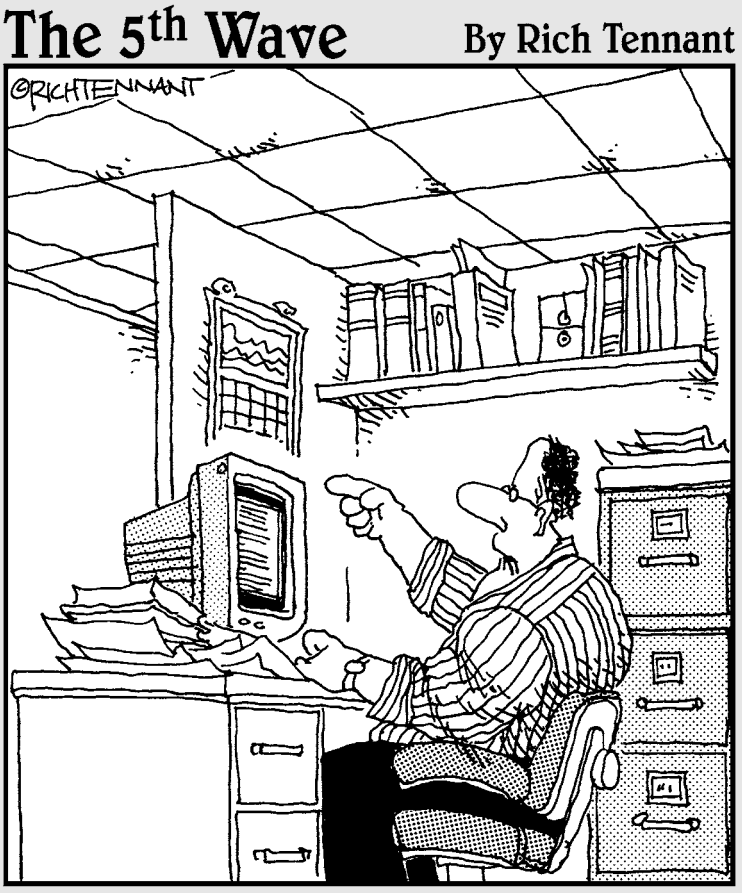
Ethical hacking isn't the be-all and end-all solution to information security. It will not guarantee security, but it's certainly a great start. Ethical hacking must be integrated as part of an overall information security program that includes

- Higher-level information risk assessments
- Strong security policies that are enforced
- Solid incident response and business continuity plans
- Effective security awareness and training initiatives

This might require hiring more staff or outsourcing more security help as well.

Don't forget about formal training for yourself and any colleagues who are helping you. You have to educate yourself consistently to stay on top of the security game.

Part VII
The Part of Tens



"Someone want to look at this manuscript I received on e-mail called 'The Embedded Virus That Destroyed the Publisher's Servers When the Manuscript was Rejected?'"

In this part . . .

Well, here's the end of the road, so to speak. In this part, I've compiled into top-ten lists what I believe are the absolute critical success factors to make ethical hacking — and information security in general — work in any organization. Bookmark, dog-ear, or do whatever you need to do with these pages so you can refer to them over and over again. This is the meat of what you need to know about information security, compliance, and managing information risks — even more so than the technical hacks and countermeasures I've covered thus far. Read it, study it, and make it happen. You can do it!

In addition, the Appendix contains a listing of my favorite ethical hacking tools and resources that I've covered, broken down into various categories for easy reference.

Chapter 19

Ten Tips for Getting Upper Management Buy-In

Dozens of key steps exist for obtaining the buy-in and sponsorship that you need to support your ethical hacking efforts. In this chapter, I describe the ones that I find are the most effective.

Cultivate an Ally and Sponsor

Selling ethical hacking and information security to management isn't something you want to tackle alone. Get an ally — preferably your direct manager or someone at that level or higher in the organization — who understands the value of ethical hacking as well as information security in general. Although this person might not be able to speak for you directly, she can be seen as an unbiased third-party sponsor and can give you more credibility.

Don't Be a FUDdy Duddy

Sherlock Holmes said, "It is a capital offense to theorize before one has data." Accordingly, it's up to you to make a good case and to put information security and the need for ethical hacking on upper management's radar. Don't blow stuff out of proportion for the sake of stirring up fear, uncertainty, and doubt (FUD). Managers worth their salt see right through that. Focus on educating management with practical advice. Rational fears proportional to the threat are fine — just don't take the Chicken Little route, claiming that the sky is falling with everything.

Demonstrate How the Organization Can't Afford to Be Hacked

Show how dependent the organization is on its information systems. Create *what-if* scenarios — sort of a business impact assessment — to show what can happen and how long the organization can go without using the network, computers, and data. Ask upper-level managers what they would do without their computer systems and IT personnel — or what they'd do if sensitive business or client information was compromised. Show real-world anecdotal evidence on hacker attacks, including malware, physical security, and social engineering issues — but be positive about it. Don't approach management negatively with FUD. Rather, keep them informed on serious security happenings. So that management can relate, find stories regarding similar businesses or industries. (A good resource is the Privacy Rights Clearinghouse listing, Chronology of Data Breaches, at `www.privacyrights.org/ar/ChronDataBreaches.htm`.) Clip magazine and newspaper articles as well. Let the facts speak for themselves.

Google is a great tool to find practically everything you need regarding information security breaches.

Show management that the organization *does* have what a hacker wants. A common misconception among those ignorant to information security threats and vulnerabilities is that their organization or network is not really at risk. Be sure to point out the potential costs from damage caused by hacking:

- Missed opportunity costs
- Loss of intellectual property
- Liability issues
- Legal costs
- Compliance-related fines
- Lost productivity
- Clean-up time and costs
- Costs of fixing a tarnished reputation

Outline the General Benefits of Ethical Hacking

In addition to the potential costs listed in the previous section, talk about how ethical hacking can help find security vulnerabilities in information systems that normally might be overlooked. Tell management that ethical hacking is a way of thinking like the bad guys so that you can protect yourself from the bad guys — Sun Tzu's "know your enemy" mindset from *The Art of War*.

Show How Ethical Hacking Specifically Helps the Organization

Document benefits that support the overall business goals:

- **Demonstrate how security can be inexpensive and can save the organization money in the long run.**
  - Security is much easier and cheaper to build in up front than to add on later.
  - Security doesn't have to be inconvenient and can enable productivity if it's done properly.
- **Discuss how new products or services can be offered for a competitive advantage if secure information systems are in place.**
  - State and federal privacy and security regulations are met.
  - Business partner and customer requirements are met.
  - Managers and the company come across as business worthy.
  - Ethical hacking shows that the organization is protecting sensitive customer and business information.
- **Outline the compliance benefits of in-depth security testing.**

Get Involved in the Business

Understand the business — how it operates, who the key players are, and what politics are involved:

- **Go to meetings to see and be seen.** This can help prove that you're concerned about the business.

- **Be a person of value who's interested in contributing to the business.**
- **Know your opposition.** Again, use the "know your enemy" mentality — if you understand what you're dealing with, buy-in is *much* easier to get.

Establish Your Credibility

Focus on these three characteristics:

- **Be positive about the organization, and prove that you really mean business.** Your attitude is critical.
- **Empathize with managers and show them that you understand the business side and what they're up against.**
- **To create any positive business relationship, you must be trustworthy.** Build that trust over time and selling security will be *much* easier.

Speak on Management's Level

No one is really that impressed with techie talk. Talk in terms of the business. This key element of obtaining buy-in is actually part of establishing your credibility but deserves to be listed by itself.

I've seen countless IT and security professionals lose upper-level managers as soon as they start speaking. A megabyte here; stateful inspection there; packets, packets everywhere! Bad idea. Relate security issues to everyday business processes and job functions. Period.

Show Value in Your Efforts

Here's where the rubber meets the road. If you can demonstrate that what you're doing offers business value on an ongoing basis, you can maintain a good pace and not have to constantly plead to keep your ethical hacking program going. Keep these points in mind:

- **Document your involvement in IT and information security, and create ongoing reports for management regarding the state of security in the organization.** Give management examples of how the organization's systems will be secured from attacks.
- **Outline tangible results as a proof of concept.** Show sample vulnerability assessment reports you've run on your systems or from the security tool vendors.

- **Treat doubts, concerns, and objections by upper management as requests for more information.** Find the answers and go back armed and ready to prove your ethical hacking worthiness.

Be Flexible and Adaptable

Prepare yourself for skepticism and rejection at first — it happens a lot, especially from upper-level managers such as CFOs and CEOs, who are often completely disconnected from IT and security in the organization.

Don't get defensive. Security is a long-term process, not a short-term product or single assessment. Start small — with a limited amount of resources, such as budget, tools, and time, and then build the program over time.

Studies have found that new ideas presented casually and without pressure are considered and have a higher rate of acceptance than ideas that are forced on people under a deadline. Just like with a spouse or colleagues at work, if you focus on and fine tune your approach — at least as much as you focus on the content of what you're going to say — you can often get people on your side, and in return, get a lot more accomplished.

Chapter 20

Ten Reasons Hacking Is the Only Effective Way to Test

Ethical hacking is not just for fun or show. For numerous business reasons, ethical hacking is the only effective way to find the security vulnerabilities that matter in your organization.

The Bad Guys Are Thinking Bad Thoughts, Using Good Tools, and Developing New Attack Methods

If you're going to keep up with external attackers and malicious insiders, you have to stay current on the latest attack methods and tools.

IT Governance and Compliance Is More Than High-Level Checklist Audits

With all the government laws and industry regulations in place, your business likely doesn't have a choice in the security matter. The problem is that being "compliant" with these laws and regulations doesn't automatically mean you're "secure." You have to take the checklist audit blinders off and dig in deeper using ethical hacking tools and techniques in order to find out what really matters.

Ethical Hacking Complements Audits and Security Evaluations

No doubt, someone in your organization understands higher-level security audits better than this ethical hacking stuff. However, if you can sell that person on ethical hacking and integrate it into existing security initiatives, the auditing process can go much deeper and improve your outcomes. Everyone wins.

Someone's Going to Ask How Secure Your Systems Are

Many businesses now require in-depth security assessments of their business partners. The same goes for certain clients. The bigger companies might want to know how secure their information is on your network. The only way to definitively know where things stand is to use the methods and tools I cover in this book.

The Law of Averages Is Working Against Businesses

Information systems are becoming more complex by the day. Literally. It's just a matter of time before these complexities work against you in the bad guys' favor. If you're going to stay informed and ensure your critical business systems and the sensitive information they process and store stay secure, you have to look at things with a malicious mindset.

Ethical Hacking Creates a Better Understanding of What the Business Is Up Against

You can say passwords are weak or patches are missing but actually exploiting such flaws and showing the outcome is quite another feat. There's no better way to prove there's a problem and motivate management to do something about it than by showing the outcomes of ethical hacking.

If a Breach Occurs, You Have Something to Fall Back On

In the event a malicious insider or external attacker still breaches your security, your business is sued, or falls out of compliance with laws or regulations, management can at least demonstrate that they were performing due diligence to uncover security risks on a periodic and consistent basis.

Ethical Hacking Brings Out the Worst in Your Systems

Someone walking around with a checklist will find security "best practices" you're missing, but they're not going to find most of the in-depth security flaws that ethical hacking is going to uncover. You know, the ones that can get you into the worst trouble. Ethical hacking brings out the warts and all.

Ethical Hacking Combines the Best of Penetration Testing and Vulnerability Testing

Penetration testing is rarely enough to find everything in your systems — the scope of traditional penetration testing is simply too limited. Neither is vulnerability testing. Ethical hacking combines the best of both and gets you the most bang for your buck.

Ethical Hacking Can Uncover Operational Weaknesses That Might Go Overlooked For Years

Ethical hacking not only uncovers technical, physical, and human weaknesses but it can also reveal problems with IT and security operations, such as patch management, change management, and lack of awareness, that may not be found otherwise.

Chapter 21

Ten Deadly Mistakes

Several deadly mistakes — when properly executed, of course — can wreak havoc on your ethical hacking outcomes and even your career. In this chapter, I discuss the potential pitfalls to be keenly aware of.

Not Getting Prior Approval in Writing

Getting documented approval, such as an e-mail, an internal memo, or a formal contract for your ethical hacking efforts — whether it's from management or your client — is an absolute must. It's your Get Out of Jail Free card.

Obtain documented approval that includes the following:

- Your plan, your schedule, and the systems to test.
- An *authorized* decision-maker's signature agreeing to the terms of your plan and agreeing not to hold you liable for malicious use or other bad things that can happen unintentionally.

No exceptions here — especially when you're doing work for clients: Make sure you get a signed copy of this document for your files.

Assuming That You Can Find All Vulnerabilities during Your Tests

So many security vulnerabilities exist — known and unknown — that you won't find them all during your testing. Don't make any guarantees that you'll find *all* the security vulnerabilities in a system. You'll be starting something that you can't finish.

Stick to the following tenets:

- Be realistic.
- Use good tools.
- Get to know your systems and practice honing your techniques.

Assuming That You Can Eliminate All Security Vulnerabilities

When it comes to computers, maintaining 100 percent, ironclad security is not attainable. You can't possibly prevent *all* security vulnerabilities, but you'll do fine if you

- Follow solid practices.
- Patch and harden your systems.
- Apply reasonable security countermeasures.

Performing Tests Only Once

Ethical hacking is a snapshot of your overall state of security. New threats and vulnerabilities surface continually, so you must perform these tests periodically and consistently to make sure you keep up with the latest security defenses for your systems.

Thinking That You Know It All

No one working with computers or information security knows it all. Keeping up with all the software versions, hardware models, and emerging technologies, not to mention the associated security threats and vulnerabilities, is impossible. Good ethical hackers know their limitations — that is, what they don't know. However, ethical hackers certainly know where to get answers. (Hint: Try Googling it.)

Running Your Tests without Looking at Things from a Hacker's Viewpoint

Think about how a malicious outsider or rogue insider can attack your network and computers. Get a fresh perspective, and try to think outside the proverbial "box." Study criminal and hacker behaviors and common hack attacks so you know what to test for.

Not Testing the Right Systems

Focus on the systems and operations that matter most. You can hack away all day at a standalone desktop running MS-DOS from a 5¼-inch floppy disk with no network card and no hard drive, but does that do any good? Probably not. But you never know. Your biggest risks might be on the seemingly least critical system. Focus on what's urgent and important.

Not Using the Right Tools

Without the right tools for the task, getting anything done without driving yourself nuts is impossible. Download the free tools I mention throughout this book and in Appendix A. Buy commercial tools when you can — they're usually worth every penny. No security tool does it all, though. Building your toolbox and getting to know your tools well will save you gobs of effort, and you'll impress others with your results.

Pounding Production Systems at the Wrong Time

One of the best ways to tick off your manager — or lose your customer's trust — is to run hack attacks against production systems when everyone is using them. If you try to hack a system at the wrong time, expect that something will take down the critical systems at the absolute worst moment. Make sure you know the best time to perform your testing. It might be in the middle of the night. (I never said ethical hacking is easy!) This might be reason to justify using security tools and other supporting utilities that can help automate certain ethical hacking tasks.

Outsourcing Testing and Not Staying Involved

Outsourcing is great, but you must stay involved throughout the entire process. Handing over the reins of your security testing to a third party without following up and staying on top of what's taking place is a bad idea. You won't be doing your manager or customers a favor by staying out of their hair. Get in their hair. (But not like a piece of chewing gum — that just makes everything more difficult.)

Appendix

Tools and Resources

To stay up to date with the latest and greatest ethical hacking tools and resources, you have to know where to turn to. This appendix contains my favorite security sites, tools, resources, and more that you can benefit from in your ongoing ethical hacking program.

This book's online Cheat Sheet contains links to all the online tools and resources listed in this appendix. Check it out at `www.dummies.com/cheatsheet/hacking`.

Bluetooth

BlueScanner — `https://labs.arubanetworks.com`

Bluesnarfer — `www.alighieri.org/tools/bluesnarfer.tar.gz`

BlueSniper rifle — `www.tomsguide.com/us/how-to-bluesniper-pt1,review-408.html`

Blooover — `http://trifinite.org/trifinite_stuff_blooover.html`

Bluejacking community site — `www.bluejackq.com`

BTScanner for XP — `www.pentest.co.uk/src/btscanner_1_0_0.zip`

Car Whisperer — `http://trifinite.org/trifinite_stuff_carwhisperer.html`

Detailed presentation on the various Bluetooth attacks — `http://trifinite.org/Downloads/21c3_Bluetooth_Hacking.pdf`

NIST Special Publication 800-48 — `http://csrc.nist.gov/publications/nistpubs/800-48-rev1/SP800-48r1.pdf`

Smurf — `www.gatefold.co.uk/smurf`

Certifications

Certified Ethical Hacker — www.eccouncil.org/CEH.htm

Certified Information Security Manager — www.isaca.org

Certified Information Systems Security Professional — www.isc2.org/cissp/default.aspx

Certified Wireless Security Professional — www.cwnp.com/cwsp/index.html

CompTIA Security+ — www.comptia.org/certifications/listed/security.aspx

SANS GIAC — www.giac.org

Databases

Advanced Access Password Recovery — www.elcomsoft.com/acpr.html

Advanced SQL Password Recovery — www.elcomsoft.com/asqlpr.html

AppDetectivePro — www.appsecinc.com/products/appdetective

Elcomsoft Distributed Password Recovery — www.elcomsoft.com/edpr.html

Microsoft SQL Server Management Studio Express — www.microsoft.com/express/sql/default.aspx

NGSSQuirreL — www.ngssoftware.com/products/database-security

Pete Finnigan's listing of Oracle scanning tools — www.petefinnigan.com/tools.htm

QualysGuard — www.qualys.com

SQLPing3 — www.sqlsecurity.com/Tools/FreeTools/tabid/65/Default.aspx

Exploit Tools

Metasploit — www.metasploit.com

Milw0rm — www.milw0rm.com

General Research Tools

AfriNIC — www.afrinic.net

APNIC — www.apnic.net

ARIN — https://ws.arin.net/whois/index.html

Bing — www.bing.com

DNSstuff.com — www.DNSstuff.com

dnstools.com — www.dnstools.com

The File Extension Source — http://filext.com

Google — www.google.com

Government domains — www.dotgov.gov

Hoover's business information — www.hoovers.com

LACNIC — www.lacnic.net

Military domains — www.nic.mil

Netcraft's *What's that site running?* — www.netcraft.com

RIPE Network Coordination Centre — www.db.ripe.net/whois

Switchboard.com — www.switchboard.com

U.S. Patent and Trademark Office — www.uspto.gov

US Search.com — www.ussearch.com

U.S. Securities and Exchange Commission — www.sec.gov/edgar.shtml

Wotsit's Format — www.wotsit.org

Whois.net — www.whois.net

Whatismyip.com — www.whatismyip.com

Yahoo! Finance — http://finance.yahoo.com

Zabasearch — www.zabasearch.com

Hacker Stuff

2600 The Hacker Quarterly — www.2600.com

Computer Underground Digest — http://cu-digest.org/

Hacker T-shirts, equipment, and other trinkets — www.thinkgeek.com

Hackin9 — http://hakin9.org

Honeypots: Tracking Hackers — www.tracking-hackers.com

The Online Hacker Jargon File — www.jargon.8hz.com

PHRACK — www.phrack.org

Keyloggers

Invisible KeyLogger Stealth — www.amecisco.com/iks.htm

KeyGhost — www.keyghost.com

SpectorSoft — www.spectorsoft.com

Laws and Regulations

Gramm-Leach-Bliley Act (GLBA) Safeguards Rule — www.ftc.gov/os/2002/05/67fr36585.pdf

Health Information Technology for Economic and Clinical Health (HITECH) Act — www.oig.dot.gov/files/Recovery_Act.pdf

Health Insurance Portability and Accountability Act (HIPAA) Security Rule — www.cms.hhs.gov/securitystandard/downloads/securityfinarule.pdf

Payment Card Industry Data Security Standard (PCI DSS) — www.pcisecuritystandards.org/security_standards/pci_dss.shtml

U.S. state breach notification laws — www.ncsl.org/programs/lis/cip/priv/breachlaws.htm

Linux

BackTrack — www.remote-exploit.org/backtrack.html

freshmeat.net — http://freshmeat.net

GFI LANguard — www.gfi.com/lannetscan

Linux Security Auditing Tool (LSAT) — http://usat.sourceforge.net

QualysGuard — www.qualys.com

SourceForge — http://sourceforge.net

THC-Amap — http://freeworld.thc.org/thc-amap

Tiger — www.nongnu.org/tiger

Live Toolkits

BackTrack — www.remote-exploit.org/backtrack.html

Comprehensive listing of live bootable Linux toolkits — www.frozentech.com/content/livecd.php

Knoppix — www.knoppix.net

Network Security Toolkit — www.networksecuritytoolkit.org

Security Tools Distribution — http://s-t-d.org

Log Analysis

ArcSight Logger — www.arcsight.com/products/products-logger

GFI EventsManager — www.gfi.com/eventsmanager

LogAnalysis.org system logging resources — www.loganalysis.org

Messaging

Abuse.net SMTP relay checker — www.abuse.net/relay.html

Brutus — www.hoobie.net/brutus

Cain & Abel — www.oxid.it/cain.html

DNSstuff.com relay checker — www.dnsstuff.com

EICAR Anti-Virus test file — www.eicar.org/anti_virus_test_file.htm

GFI e-mail security test — www.gfi.com/emailsecuritytest

mailsnarf — www.monkey.org/~dugsong/dsniff or

smtpscan — www.freshports.org/security/smtpscan

Miscellaneous Tools

FreeZip — http://members.ozemail.com.au/~nulifetv/freezip

WinZip — www.winzip.com

NetWare

Craig Johnson's BorderManager resources — http://nscsysop.hypermart.net

JRB Software — www.jrbsoftware.com

NetServerMon — www.simonsware.com/nsmdesc.html

Pandora — www.nmrc.org/project/pandora

Rcon program — http://packetstormsecurity.nl/Netware/penetration/rcon.zip

Remote — www.securityfocus.com/data/vulnerabilities/exploits/Remote.zip

UserDump — www.hammerofgod.com/download/userdump.zip

Networks

Arpwatch — http://linux.maruhn.com/sec/arpwatch.html

Blast — www.foundstone.com/us/resources/proddesc/blast.htm

Cain & Abel — www.oxid.it/cain.html

CommView — www.tamos.com/products/commview

dsniff — www.monkey.org/~dugsong/dsniff

Essential NetTools — www.tamos.com/products/nettools

ettercap — http://ettercap.sourceforge.net

Firewalk — www.packetstormsecurity.org/UNIX/audit/firewalk

Getif — www.wtcs.org/snmp4tpc/getif.htm

GFI LANguard — www.gfi.com/lannetscan

IETF RFCs — www.rfc-editor.org/rfcxx00.html

IKEcrack — http://ikecrack.sourceforge.net

MAC address vendor lookup — http://standards.ieee.org/regauth/oui/index.shtml

MAC Changer — www.alobbs.com/macchanger

Nessus vulnerability scanner — www.nessus.org

Netcat — http://netcat.sourceforge.net

Netfilter/iptables — www.netfilter.org

NetResident — www.tamos.com/products/netresident

NetScanTools Pro — www.netscantools.com

Nmap port scanner — http://nmap.org

NMapWin — http://sourceforge.net/projects/nmapwin

OmniPeek — www.wildpackets.com/products/distributed_network_analysis/omnipeek_network_analyzer

Port number listing — www.iana.org/assignments/port-numbers

Port number lookup — www.cotse.com/cgi-bin/port.cgi

PortSentry — http://sourceforge.net/projects/sentrytools

PromiscDetect — http://ntsecurity.nu/toolbox/promiscdetect

QualysGuard vulnerability scanner — www.qualys.com

SMAC MAC address changer — www.klcconsulting.net/smac

SNARE — www.intersectalliance.com/projects/Snare

sniffdet — http://sniffdet.sourceforge.net

SNMPUTIL — www.wtcs.org/snmp4tpc/FILES/Tools/SNMPUTIL/SNMPUTIL.zip

SuperScan port scanner — www.foundstone.com/us/resources/proddesc/superscan.htm

TCP Wrappers — http://itso.iu.edu/TCP_Wrappers

TrafficIQ Pro — www.karalon.com

UDPFlood — www.foundstone.com/us/resources/proddesc/udpflood.htm

WhatIsMyIP — www.whatismyip.com

Wireshark — www.wireshark.org

Password Cracking

Advanced Archive Password Recovery — www.elcomsoft.com/archpr.html

BIOS passwords — http://labmice.techtarget.com/articles/BIOS_hack.htm

Brutus — www.hoobie.net/brutus

Cain & Abel — www.oxid.it/cain.html

Crack — ftp://coast.cs.purdue.edu/pub/tools/unix/pwdutils/crack

Default vendor passwords — www.cirt.net/passwords

Dictionary files and word lists

ftp://ftp.cerias.purdue.edu/pub/dict

ftp://ftp.ox.ac.uk/pub/wordlists

http://packetstormsecurity.nl/Crackers/wordlists

www.outpost9.com/files/WordLists.html

http://rs159.rapidshare.com/files/184075601/BlackKnightList.rar

Elcomsoft Distributed Password Recovery — www.elcomsoft.com/edpr.html

Elcomsoft System Recovery — www.elcomsoft.com/esr.html

John the Ripper — www.openwall.com/john

ophcrack — http://ophcrack.sourceforge.net

Pandora — www.nmrc.org/project/pandora

Password Safe — http://passwordsafe.sourceforge.net

Proactive Password Auditor — www.elcomsoft.com/ppa.html

Proactive System Password Recovery — www.elcomsoft.com/pspr.html

pwdump3 — www.openwall.com/passwords/dl/pwdump/pwdump3v2.zip

NetBIOS Auditing Tool — www.securityfocus.com/tools/543

NIST Guide to Enterprise Password Management — http://csrc.nist.gov/publications/drafts/800-118/draft-sp800-118.pdf

NTAccess — www.mirider.com/ntaccess.html

RainbowCrack — http://project-rainbowcrack.com

Rainbow tables — http://rainbowtables.shmoo.com

SQLPing3 — www.sqlsecurity.com/Tools/FreeTools/tabid/65/Default.aspx

TSGrinder — www.hammerofgod.com/download/tsgrinder-2.03.zip

WinHex — www.winhex.com

Patch Management

BigFix Patch Management — www.bigfix.com/content/patch-management

Debian Linux Security Alerts — www.debian.org/security

Ecora Patch Manager — www.ecora.com/ecora/products/patchmanager.asp

GFI LANguard — www.gfi.com/lannetscan

Linux Kernel Updates — www.linuxhq.com

Lumension Patch and Remediation — www.lumension.com/vulnerability-management/patch-management-software.jsp

Novell Patches and Security — http://support.novell.com/patches.html

Microsoft TechNet Security Center — http://technet.microsoft.com/en-us/security/default.aspx

Red Hat Linux Security Alerts — http://updates.redhat.com

Slackware Linux Security Advisories — www.slackware.com/security

SUSE Linux Security Alerts — www.novell.com/linux/download/updates/

Windows Server Update Services from Microsoft — www.microsoft.com/windowsserversystem/updateservices/default.mspx

Security Education and Learning Resources

Kevin Beaver's information security articles, whitepapers, webcasts, podcasts, and screencasts — www.principlelogic.com/resources.html

Kevin Beaver's *Security On Wheels* information security audio programs — http://securityonwheels.com

Kevin Beaver's *Security On Wheels* blog — http://securityonwheels.com/blog

Kevin Beaver's Twitter page — www.twitter.com/kevinbeaver

Security Methods and Models

Open Source Security Testing Methodology Manual — www.isecom.org/osstmm

OWASP www.owasp.org

Secur*IT*ree — www.amenaza.com

Software Engineering Institute's OCTAVE methodology — www.cert.org/octave

Source Code Analysis

Checkmarx — www.checkmarx.com

Fortify Software — www.fortifysoftware.com

Klocwork — www.klocwork.com

Ounce Labs — www.ouncelabs.com

Storage

CHAP Password Tester — www.isecpartners.com/tools.html#CPT

CIFSShareBF — www.isecpartners.com/SecuringStorage/CIFShareBF.zip

Effective File Search — www.sowsoft.com/search.htm

FileLocator Pro — www.mythicsoft.com/filelocatorpro

GFI LANguard — www.gfi.com/lannetscan

Google Desktop — http://desktop.google.com

GrabiQNs — www.isecpartners.com/SecuringStorage/GrabiQNs.zip

Identity Finder — www.identityfinder.com

NASanon — www.isecpartners.com/SecuringStorage/NASanon.zip

StorScan — www.isecpartners.com/tools.html#StorScan

SuperScan — www.foundstone.com/us/resources/proddesc/superscan.htm

System Hardening

Bastille Linux Hardening Program — http://bastille-linux.sourceforge.net

Center for Internet Security Benchmarks — www.cisecurity.org

Deep Freeze — www.faronics.com/html/deepfreeze.asp

Fortres 101 — www.fortresgrand.com

How to disable SMTP relay on various e-mail servers — www.mail-abuse.com/an_sec3rdparty.html

Imperva — www.imperva.com/products/database-firewall.html

Linux Administrator's Security Guide — www.seifried.org/lasg

PGP Whole Disk Encryption — www.pgp.com/products/wholediskencryption

Pyn Logic — www.pynlogic.com/enzoinfo2.aspx

SecureIIS — www.eeye.com/html/products/secureiis/index.html

ServerDefender — www.port80software.com/products/serverdefender

TrueCrypt — www.truecrypt.org

User Awareness and Training

Awareity MOAT — www.awareity.com

Dogwood Management Partners Security Posters — www.securitposters.net

Greenidea Visible Statement — www.greenidea.com

Interpact, Inc. Awareness Resources — www.thesecurityawarenesscompany.com

Managing an Information Security and Privacy Awareness and Training Program by Rebecca Herold (Auerbach) — www.amazon.com/Managing-Information-Security-Awareness-Training/dp/0849329639

NIST Awareness, Training, & Education resources — http://csrc.nist.gov/ATE

Security Awareness, Inc. — www.securityawareness.com

Voice over IP

Cain & Abel — www.oxid.it/cain.html

CommView — www.tamos.com/products/commview

Listing of various VoIP tools — www.voipsa.org/Resources/tools.php

NIST's SP800-58 document — http://csrc.nist.gov/publications/nistpubs/800-58/SP800-58-final.pdf

OmniPeek — www.wildpackets.com/products/distributed_network_analysis/omnipeek_network_analyzer

PROTOS — www.ee.oulu.fi/research/ouspg/protos

sipsak — http://sipsak.org

SiVuS — http://vopsec.net/html/tools.html

vomit — http://vomit.xtdnet.nl

VoIP Hopper — http://voiphopper.sourceforge.net

Vulnerability Databases

Common Vulnerabilities and Exposures — http://cve.mitre.org

CWE/SANS Top 25 Most Dangerous Programming Errors — www.sans.org/top25errors

National Vulnerability Database — http://nvd.nist.gov

Privacy Rights Clearinghouse's *A Chronology of Data Breaches* — www.privacyrights.org/ar/ChronDataBreaches.htm

SANS Top 20 Internet Security Problems, Threats, and Risks — www.sans.org/top20

US-CERT Vulnerability Notes Database — www.kb.cert.org/vuls

Wireless Vulnerabilities and Exploits — www.wve.org

Web Applications

Absinthe — www.0x90.org/releases/absinthe

Acunetix Web Vulnerability Scanner — www.acunetix.com

Brutus — www.hoobie.net/brutus/index.html

Defaced Web sites — http://zone-h.org/archive

HTTrack Website Copier — www.httrack.com

Firefox Web Developer — http://chrispederick.com/work/web-developer

Foundstone's Hacme Tools — www.foundstone.com/us/resources-free-tools.asp

Google Hack Honeypot — http://ghh.sourceforge.net

Google Hacking Database — http://johnny.ihackstuff.com/ghdb

NGSSquirrel — www.ngssoftware.com/software.htm

N-Stealth Web Application Security Scanner — www.nstalker.com/eng/products/nstealth

Paros Proxy — www.parosproxy.org

Port 80 Software's ServerMask — www.port80software.com/products/servermask

SiteDigger — www.foundstone.com/us/resources/proddesc/sitedigger.htm

SWFScan — https://h30406.www3.hp.com/campaigns/2009/wwcampaign/1-5TUVE/index.php?key=swf

WebInspect — www.spidynamics.com/products/webinspect/index.html

WebGoat — www.owasp.org/index.php/Category:OWASP_WebGoat_Project

WSDigger — www.foundstone.com/us/resources/proddesc/wsdigger.htm

WSFuzzer — www.owasp.org/index.php/Category:OWASP_WSFuzzer_Project

Windows

DumpSec — www.systemtools.com/somarsoft/?somarsoft.com

GFI LANguard — www.gfi.com/lannetscan

Microsoft Baseline Security Analyzer — `www.microsoft.com/technet/security/tools/mbsahome.mspx`

Network Users — `www.optimumx.com/download/netusers.zip`

QualysGuard — `www.qualys.com`

Sysinternals — `http://technet.microsoft.com/en-us/sysinternals/default.aspx`

Winfo — `www.ntsecurity.nu/toolbox/winfo`

Wireless Networks

Aircrack — `http://aircrack-ng.org`

AirMagnet WiFi Analyzer — `www.airmagnet.com/products/wifi_analyzer`

AirSnort — `http://airsnort.shmoo.com`

Asleap — `http://asleap.sourceforge.net`

Cantenna war-driving kit — `http://mywebpages.comcast.net/hughpep`

CommView for Wi-Fi — `www.tamos.com/products/commwifi`

Digital Hotspotter — `www.canarywireless.com`

Elcomsoft Wireless Security Auditor — `www.elcomsoft.com/ewsa.html`

Homebrew WiFi antenna — `www.turnpoint.net/wireless/has.html`

KisMAC — `http://trac.kismac-ng.org`

Kismet — `www.kismetwireless.net`

NetStumbler — `www.netstumbler.com`

OmniPeek — `www.wildpackets.com/products/omni/overview/omnipeek_analyzers`

SeattleWireless Hardware Comparison page — `www.seattlewireless.net/index.cgi/HardwareComparison`

Super Cantenna — `www.cantenna.com`

Wellenreiter — `http://sourceforge.net/projects/wellenreiter/`

WEPCrack — `http://wepcrack.sourceforge.net`

WiGLE database of wireless networks — `www.wigle.net`

WifiMaps — `www.wifimaps.com`

WiFinder — `www.wifinder.com`

WildPackets' OmniPeek — `www.wildpackets.com/products/distributed_network_analysis/omnipeek_network_analyzer`

WinAirsnort — `http://winairsnort.free.fr`

Index

• Numerics •

802.11 encryption protocols, 161–162
802.11i encryption protocols, 165

• A •

Absinthe, 363
Abuse.net SMTP relay checker, 355
access control, 217
access points (APs)
  MAC address of, 156
  network vulnerabilities and, 152
  rogue wireless devices, 165
account enumeration attacks, 257–260
account lockout, 112
Active Directory database, 93
Acunetix Web Vulnerability Scanner, 278, 288, 290, 296, 363
Address Resolution Protocol (ARP), 140
Advanced Access Password Recovery, 307, 352
Advanced Archive Password Recovery, 102–103, 358
Advanced Encryption Standard (AES), 162, 165
Advanced SQL Password Recovery, 304, 306, 352
AES (Advanced Encryption Standard), 162, 165
AfriNIC, 50, 353
Aircrack, 154, 161, 365
AirMagnet Handheld Analyzer, 155
AirMagnet WiFi Analyzer, 154, 166, 168–169, 365
Airodump, 161
AirSnort, 365
Akin, Thomas (Southeast Cybercrime Institute), 251
allintitle Google operator, 282
Amap, 215
Andrews, Chip (Special Ops Security), 305–306
anonymity, 34
Apache Web server, 209
APNIC, 51, 353
AppDetectivePro, 308, 352
application attacks, 16
Arcsight Logger, 331, 355
ARIN, 51, 353
ARP (Address Resolution Protocol), 140
ARP spoofing. *See also* network infrastructure attacks
  countermeasures, 144
  defined, 140
  how it works, 140–141
  using Cain & Abel, 141–143
Arpwatch, 144, 356
Asleap, 164, 365
Asterisk, 68
Athena FirewallGrader, 133
attack tree analysis, 39
attacks
  account enumeration attacks, 257–259
  application attacks, 16
  ARP spoofing, 140–143
  banner grabbing, 130–131, 255–257
  brute-force attacks, 95–96
  buffer overflows, 283–284
  code injection, 287–289
  database attacks, 303–309
  denial of service attacks, 145–147
  dictionary attacks, 94–95
  directory reversal attacks, 280–283
  dumpster diving, 15
  e-mail attacks, 252–267
  e-mail bombs, 252–255
  e-mail header disclosures, 263
  e-mail traffic capture, 264
  encrypted traffic, 160–165
  hidden field manipulation, 285–286
  input filtering attacks, 283–291
  instant messaging, 267–270, 287–289
  keystroke logging, 103–104

MAC address spoofing, 143–144, 170–175
malware, 264
network infrastructure attacks, 15
nontechnical, 14–15
operating system attacks, 15
password cracking, 89–109
physical, 15
rainbow attacks, 96
rconsole attacks, 233–236
reasons for, 31
SMTP attacks, 257–265
SMTP relay attacks, 260–262
social engineering, 15
SQL injection, 287–289
storage system attacks, 309–313
styles of, 32
timing, 33
URL manipulation, 285
voice over IP, 270–276
vulnerability and, 33
auditing, security, 12
Auditory Professional, 269
authenticated scans, 205–206
authorization, 18
automated assessment, 56
Awareity MOAT, 362

• B •

background checks, 49
BackTrack
capturing e-mail traffic address with, 258
firewall rulebase testing with, 133
Linux security testing with, 154, 209
network vulnerability testing with, 148
Web site, 355
banner grabbing. *See also* network infrastructure attacks
countermeasures, 131
defined, 130
overview, 130
telnet, 130–131
banners, 130, 255–257
Bastille Linux Hardening Program, 361
Beaver, Kevin (*Security On Wheels*), 360
believability in social engineering, 69
Berkeley Software Distribution (BSD) r-commands, 218–220
BigFix Patch Management, 227, 325, 359
Bing search engine, 48, 353
BIOS passwords, 107, 358
BitLocker, 198
black hat hackers, 10
BlackKnightList, 95, 296
blank password, 101–102
Blast tool, 146, 356
Blaster worm, 181
blind assessment, 42, 47
blind SQL injection, 287
Blooover, 160
Bluejacking, 160, 351
BlueScanner, 160, 351
Bluesnarfer, 160, 351
BlueSniper rifle, 160
Bluetooth, 160, 351
BorderManager resources, 356
broadcast mode, 140
brute-force attacks, 95–96
Brutus
brute-force testing with, 296
password cracking with, 93
POP3 password cracking with, 265
Web site, 355, 358, 363
BTScanner for XP, 160, 351
buffer overflows, 223–224, 283–284
building infrastructure, physical security, 78–79
business phones, 68
buy-in, management, 339
ally and sponsor, 337
benefits of ethical hacking, 339
establishing credibility, 340
flexibility and adaptability, 341
getting involved in business, 339–340
practical advice, 337
speaking on management's level, 340
value in efforts, 340–341
what-if-scenarios, 338

• C •

Cain & Abel. *See also* software and testing tools
ARP spoofing with, 141–143

capturing and recording voice traffic with, 274–276
capturing e-mail traffic with, 264
cracking Oracle password hashes with, 307
network analysis with, 105, 121, 135
password cracking with, 92, 304
Web site, 356, 358, 362
Camasia Studio, 41
Canary Wireless, 155
cantenna, 155, 365
CAPTCHA, 255, 297
Car Whisperer, 160, 351
Carrier Sense Multiple Access/Collision Avoidance (CSMA/CA) protocol, 175
case studies
database hacking, 305–306
e-mail attacks, 251
messaging-system attacks, 251
network infrastructure attacks, 118
password cracking, 87
physical security, 77
social engineering, 63
Web application attacks, 279
wireless network attack, 153
CCTV security camera, 27
Center for Internet Security, 113, 321, 326, 361
certifications, 352
Certified Ethical Hacker (CEH), 12
`.cgi` extension, 299
CHAP Password Tester, 310, 361
Chappell, Laura, 118
Character Generator pot (NetWare), 232
chargen, 123
Checkmarx, 300, 361
CheckPoint, 300
`chkconfig`, 217
chknull (password-cracking software), 92
ChoicePoint, 49
Chronology of Data Breaches, 338, 363
CIFShareBF, 310, 361
CipherTrust IronMail, 255
Cisco Global Exploiter, 148–149
civil liberties, 32
Clear Channel Assessment attack, 175–176
cleartext packets, 242
client notification, 37
closed-circuit television (CCTV), 81
code injection, 287–289
Common Vulnerabilities and Exposures, 55, 363
CommView
denial of service testing with, 146
network analysis with, 106, 135
Web site, 357, 362
CommView for Wi-Fi, 166–167, 365
Computer Underground Digest, 354
contingency plan, 18
COPS, 223
copy rooms, 81
Counter Mode with Cipher Block Chaining Message Authentication Code (CCMP), 162
countermeasures
account enumeration attacks, 257–260
ARP poisoning, 144
ARP spoofing, 144
banner attacks, 257
banner grabbing, 131
buffer overflows, 223–224
database attacks, 308–309
default configuration settings exploits, 177–178
default script attacks, 294
denial of service attacks, 146–147, 176
directory reversal attacks, 282–283
e-mail attachment attacks, 253
e-mail attacks, 266–267
e-mail bombs, 253–254
e-mail connection attacks, 253
e-mail header disclosures, 263
encrypted traffic attacks, 164–165
file permission attacks, 222–223
firewalls, 133–134
`hosts.equiv` file attacks, 219–220
input filtering attacks, 291
instant messaging vulnerabilities, 268–269
MAC address spoofing, 144, 175
missing patch exploitation, 205
NetBIOS, 190
Netware intruders, 238
NetWare Loadable Module, 241
network analyzers, 139–140
network infrastructure attacks, 135
NFS attacks, 221
null sessions, 194–195
packet capture, 242

countermeasures *(continued)*
password cracking, 109–114
physical security attacks, 224–225
physical security problems, 176–177
port scanning, 127–128
`.rhost` file attacks, 219–220
rogue NLM attack, 241
rogue wireless devices, 170
SMTP relay attacks, 263
SNMP scanning, 130
social engineering, 72–74
storage system attacks, 313
system scans, 212
unneeded services, 216–217
unsecured login mechanisms, 297
Voice over IP, 276
vulnerable wireless workstations, 177–178
wireless network attacks, 158–159
Counterpane, 331
Crack, 358
crackers, 10
cracking tools, 20
cracklib, 114
crashing system during tests, 17
criminal hackers, 10, 28
cross-site scripting (XSS), 290
CWE/SANS Top 25 Most Dangerous Programming Errors, 363
CxAudit, 300
CxDeveloper, 300–301
cyberterrorists, 29

• D •

daemons, 209
Data Thief tool, 305
database attacks. *See also* storage system attacks
best practices for minimizing risks, 308–309
case study, 305–306
finding databases on network, 304–306
overview, 303
password cracking, 306–307
scanning for vulnerabilities, 307–308
testing tools, 303–304
tools, 352
daytime, 123
`.db` file, 311
`.dbf` file, 311
Debian Linux Security Alerts, 359
Debian Package System, 227
Deep Freeze, 104, 361
defaced Web pages, 31, 364
default configuration settings, 178
default script attacks, 292–293
deliverables, 19
denial of service (DoS) attacks. *See also* network infrastructure attacks
countermeasures, 146–147
defined, 145
distributed, 145
Ping of Death, 145
Queensland, 175–176
SYN floods, 145
testing, 146
WinNuke, 145
dictionary attacks, 94–95
dictionary files, 358
Digital Hotspotter, 155, 168, 365
directional antenna, 155
directory reversal attacks
countermeasures, 282–283
crawlers, 280–281
defined, 280
Google, 281–282
distributed denial of service (DDoS) attacks, 145
D-Link DWL-650 wireless NIC, 175
DNS (Domain Name System), 123
DNSstuff.com, 50, 353
dnstools.com, 353
`.doc` file, 311
`.docx` file, 311
Dogwood Management Partners Security Posters, 362
Domain Name System (DNS), 123
doors, 79–80
Draper, John (Captain Crunch), 27
drop ceilings, 80
dsniff program, 140, 357
dsrepair (NetWare Loadable Module), 239
DumpSec utility, 55, 192, 364
dumpster diving, 15, 67

• E •

eBlaster (keystroke-logging software), 104
echo, 123
Echo port (NetWare), 232
Ecora Patch Manager, 326, 359
eDirectory, 229
Effective File Search, 310, 361
EICAR Anti-Virus test file, 356
EICAR test string, 265
Elcomsoft Advanced Archive Password Recovery, 102–103
Elcomsoft Distributed Password Recovery, 92, 306, 352, 358
Elcomsoft System Recovery, 108–109, 359
Elcomsoft Wireless Security Auditor (EWSA), 154, 162–163, 365
e-mail attacks. *See also* messaging-system attacks
  banners, 255–257
  case study, 251
  e-mail bombs, 252–255
  guidelines against, 266–267
  overview, 16
  SMTP attacks, 257–265
  software solutions against, 266
e-mail bombs. *See also* messaging-system attacks
  automated security controls against, 254–255
  bandwidth blocking, 253–254
  countermeasures against connection attacks, 253–254
  perimeter protection, 255
  storage overload, 253
  using attachments, 252–253
  using connections, 253–254
  using floods of e-mails, 253
e-mail firewalls, 255
e-mail header disclosures, 263–264
e-mail security testing zone, 265
e-mail servers, 55
e-mail traffic, capturing, 264
EMail Verify, 257–258
encrypted traffic, 160–164
encrypted traffic attacks. *See also* wireless network attacks
  countermeasures, 164–165
  encryption protocols, 161–162
  overview, 161–162
  tools, 162–164
enumeration utility, 55, 185–186
error-based SQL injection, 287
errors and omission insurance, 35
Essential NetTools, 120, 357
ethical hackers, 10
ethical hacking
  assessing vulnerabilities, 55–57
  attack tree analysis, 39
  vs. auditing, 12
  automating, 329–330
  avoiding system crashes in, 17
  blind assessment in, 47
  certification, 12
  compliance and regulatory concerns, 12–13
  defined, 11
  determining systems to hack, 37–40
  evaluating results in, 22–23
  executing plan in, 22
  footprinting, 47–52
  formulating plan, 18–19
  gathering public information, 48–49
  goals, 13–14, 36–37
  insurance, 35
  logging information in, 46
  mistakes in, 347–350
  network mapping, 50–52
  outsourcing, 332–333
  penetrating system, 57–58
  performing, 45–47
  policy considerations, 12
  reasons for effectiveness of, 343–345
  respecting privacy in, 17
  scanning systems, 52–53
  similarity to beta testing, 45
  similarity to malicious attacks, 46
  testing standards, 40–43
  tools, 20–21, 44
  working ethically, 16–17
ettercap utility, 135, 357
Event ID 4226 Patcher tool, 185
event logging system, 331
EventsManager, 331
exploit tools, 352

EXPN command, 257, 259
external attachers, 10

• F •

Facebook, 145
Fedora Linux, 154
File Extension Source, The, 353
file permission attacks, 221–223
file sharing, 267–268
File Transfer Protocol (FTP), 123, 209, 213
FileLocator Pro, 309, 310, 312, 361
filetype:file-extension hostname: query (Google), 281
findstr, 104
finger, 123
Finnigan, Pete, 352
fire detection and suppression systems, 79
Firefox, configuring for Web proxy, 284
Firefox Web Developer, 278, 284, 298, 364
Firewalk, 133, 357
firewalls
  countermeasures against attacks, 133–134
  e-mail, 255
  testing, 131–134
  Web security, 299–300
Flash files, 48–49
Fluke WiFi Analyzer, 154
footprinting. *See also* ethical hacking
  gathering public information, 47–49
  overview, 47
  Web crawling, 49
  Web search, 48
Fortify Software, 361
Fortres program 101, 104, 361
Foundstone, 298, 364
fping, 52
FreeZip, 97, 356
freshmeat.net, 209
FTP (File Transfer Protocol), 123, 209, 213
FTP control, 123
fully qualified domain names (FQDNs), 51

• G •

Getif utility, 120, 128, 232, 357
GFI e-mail security test, 356
GFI EventsManager, 355
GFI LANguard. *See also* software and testing tools
  authenticated scans with, 206
  Linux system testing with, 208
  NetWare vulnerability testing with, 230, 233
  patch automation with, 326
  share finder, 197
  storage system testing with, 309
  system scanning with, 186, 210–212
  vulnerability assessment with, 121
  Web site, 357, 359, 361, 364
  Windows system testing with, 184
goals in ethical hacking, 36–37
Goog Mail Enum, 258, 260
Google, 20, 48, 67, 281–282, 353
Google Desktop, 361
Google Groups, 51, 282
Google Hacking Database (GHDB), 282, 364
Google Hacking for Penetration Testers (Long), 282
government domains, 353
GrabiQNs, 310, 361
Gramm-Leach-Bliley Act (GLBA), 13, 354
Greenidea Visible Statement, 362
grep, 104
GroupWise (NetWare), 232

• H •

hackers
  anonymity, 34
  behavior of, 26
  black hat, 10
  categories of, 28
  criminal hackers, 28
  cyberterrorists, 29
  defined, 10
  ethical, 10
  hackers for hire, 29
  hacktivists, 29
  mindset of, 27
  motivations of, 25–26, 29–31
  online resources, 354
  reformed, 333
  script kiddies, 26, 28

security researchers, 28
stereotypical view of, 25–26
white hat, 10
Hackin9, 33, 354
hacking
civil liberties and, 32
planning and performing, 32–33
reasons for, 29–31
vulnerabilities in security and, 33
Hacking Wireless Networks For Dummies (Davis), 162, 166, 321
hacktivists, 29
Hacme Tools, 298, 364
hardening, 326–327, 361–362
Health Information Technology for Economic and Clinical Health (HITECH), 354
Health Insurance Portability and Accountability Act (HIPAA), 12–13, 354
hidden field manipulation, 285–286
high-impact vulnerabilities, 320
Homebrew WiFi antenna, 365
Honeypots: Tracking Hackers, 354
Hoover's business information, 353
hosting providers, notifying, 41
hosts, scanning, 52
`hosts.equiv` file attacks, 218–220
HTTP (Hypertext Transfer Protocol), 123
HTTP Get requests, 292
HTTP POST requests, 292
HTTP proxy, 123
HTTPS (HTTP over SSL), 123
HTTrack Website Copier, 49, 278, 280–281, 364
HyperTerminal, 261
Hypertext Transfer Protocol (HTTP), 16, 123

• I •

ICMP (Internet Control Message Protocol), 123–124
Identity Finder, 309, 312–313, 361
Identity Finder Pro, 104
IKE (Internet Key Exchange), 148–149
IKEcrack, 357
Imperva, 362
`inetd.conf`, 216–217
inference, 90–91
information-gathering
overview, 47
port scanning, 53–54
system scans, 52–53, 209–212
Web crawling, 49
Web search, 48–49
Web sites, 49
InGuardians, Inc., 153
input filtering attacks. *See also* Web application attacks
buffer overflows, 283–284
code injection, 287–289
countermeasures, 291
cross-site scripting, 290
hidden field manipulation, 285–286
overview, 283
SQL injection, 287–289
URL manipulation, 285
(IN)SECURE Magazine, 33
instant messaging
countermeasures against vulnerabilities, 268–269
detecting traffic, 269
log files, 268
overview, 267
sharing files in, 267–268
system configuration, 269–270
user behavior, 269
vulnerabilities, 267–268
insurance, 35
Internet Control Message Protocol (ICMP), 123–124
Internet Key Exchange (IKE), 148–149
Internet Security Advisors Group, 62
Internet service providers (ISP), notifying, 41
Internet services, 209
intruder lockout, 112
intrusion detection, 237–238
intrusion prevention system, evading, 27
inurl Goole operator, 282
Invisible KeyLogger Stealth, 104, 354
IP address, scanning, 52
IP Personality, 299
IP spoofing, 147

• J •

JavaScript, 290
John the Ripper
  cracking Unix passwords with, 98
  cracking Windows passwords with, 96–98
  overview, 92
  Web site, 359
Johnson, Craig, 356
JRB Software, 240, 356
Juniper Networks, 300, 313

• K •

Karalon, 132
Kerberos, 91
KeyGhost, 104, 354
keypads, programmable, 81
keys, 81
keystroke logging, 103–104, 354
KisMAC, 161, 365
Kismet, 154, 166
KLC Consulting, 173
Klockwork, 300, 361
Knoppix Linux, 108, 154, 355
knowledge assessment, 42
Korean National Police Agency, 29

• L •

L0phtrcrack, 99
LACNIC, 51, 353
LANguard. *See also* software and testing tools
  authenticated scans with, 206
  Linux system testing with, 208
  NetWare vulnerability testing with, 230, 233
  patch automation with, 326
  share finder, 197
  storage system testing with, 309
  system scanning with, 186, 210–212
  vulnerability assessment with, 121
  Web site, 357, 359, 361, 364
  Windows system testing with, 184
laptops, locking, 84
laws and regulations, 354
LEAP protocol, 164
likability in social engineering, 69
link Goole operator, 282
LinkedIn, 20
Linux Administrator's Security Guide., 362
Linux Kernel Updates, 359
Linux operating system
  attacks, 15
  distribution updates, 227
  multiplatform update managers, 227
  overview, 207–208
  password protection in, 114
  password storage location in, 94
  patching, 227
  reasons for popularity of, 207
  unneeded services, 213–218
Linux Security Auditing Tool (LSAT), 209
Linux systems. *See also* Windows systems
  buffer overflows, 223–224
  file permission attacks, 221–223
  general security tests, 225–226
  `hosts.equiv` file attacks, 218–220
  multiplatform update managers, 227
  NFS attacks, 220–221
  overview, 207
  patching, 226
  physical security attacks, 224–225
  `.rhosts` file attacks, 218–220
  security tools, 208–209
  system scanning, 209–212
  vulnerabilities, 208
  websites, 355
live toolkits, 355
location of testing, 43
lockdown programs, 104
log analysis, 355
LogAnalysis.org, 355
Logger, 331
LoveBug worm, 72
low-impact vulnerabilities, 320
`lsof`, 215
Lumension Patch and Remediation, 227, 359

• M •

M+Guardian, 255
MAC (Media Access Control), 140, 170
MAC address, 156
MAC address spoofing. *See also* wireless network attacks
  countermeasures, 144, 175
  overview, 170–171
  steps in, 170–174
  Unix-based systems, 143
  Windows systems, 143–144
MAC address vendor lookup, 357
MAC Changer, 173, 357
MafiaBoy (hacker), 145
magazines, 33
mail rooms, 81
Mailsnarf, 264, 356
malicious internal users, 10
malicious users
  defined, 10
  monitoring, 330–331
malware, 264–266
managed security services provider (MSSP), 331
manual assessment, 56
MD5 passwords, 114
Media Access Control (MAC), 140, 170
medium-impact vulnerabilities, 320
Message Architect, 255
messaging-system attacks
  case study, 251
  e-mail attacks, 252–267
  instant messaging, 267–270
  testing tools, 355–356
  voice over IP, 270–276
  vulnerabilities, 249–250
  Web sites, 355–356
Metasploit, 58, 184, 199–205, 265, 352
Microsoft Access database files, 307
Microsoft Baseline Security Analyzer (MBSA), 183, 206, 326, 365
Microsoft Exchange, 256
Microsoft IIS server, 299
Microsoft PPTP VPN, 123
Microsoft SQL Monitor, 123
Microsoft SQL Server, 123
Microsoft SQL Server Management Studio Express, 352
Microsoft TechNet Security Center, 360
Microsoft Update, 326
military domains, 353
Milw0rm, 352
mirroring, 278
missing patch exploitation
  countermeasures, 205
  overview, 198
  using Metasploit, 199–205
mistakes in ethical hacking, 350
Mitnick, Kevin, 27
monitor mode, 137
multiplatform update managers, 227

• N •

NAP (Network Access Protection), 198
NASanon, 310, 361
National Institute of Standards and Technology (NIST), 88, 326
National Vulnerability Database, 88, 213, 321
`nbstat`, 183, 188–189
Nessus, 121, 209, 357
`net view` command, 192
NetBIOS (Network Basic Input/Output System)
  countermeasures, 190
  hacks, 188
  overview, 187
  shares, 189–190
  unauthenticated enumeration, 188–189
  vulnerable ports, 188
NetBIOS Auditing Tool, 359
NetBios over TCP/IP, 123
Netcat, 132, 357
Netcraft, 54, 353
Netfilter/iptables, 357
NetResident, 134, 264, 268, 357
NetScanTools Pro
  denial of service testing with, 146
  Linux system testing with, 212
  network analysis with, 120
  port scanning with, 126–127
  system scanning with, 52
  Web site, 357
NetScreen, 300, 313
NetServerMon, 356

netstat, 183, 215
NetStumbler, 154, 157–158, 166–167, 365
NetUsers, 193
NetWare Administrator, 244
NetWare Core Protocol, 232
NetWare Loadable Module (NLM)
  admin utilities, 240
  countermeasures, 241
  documentation, 241
  dsrepair, 239
  modules command, 238–239
  overview, 234
  setpwd password reset tool, 238
  tcpcon, 239–240
  unauthenticated logins, 241
network
  countermeasures, 84
  Internet Key Exchange weaknesses, 148
  physical attacks, 82–83
  unsecured interfaces, 147–148
  vulnerabilities, 83, 147–149
Network Access Protection (NAP), 198
network analyzers, 53
  Cain & Abel, 135
  CommView, 135
  configuring, 137
  countermeasures, 139–140
  defined, 134
  detecting, 140
  ettercap, 135
  functions of, 134
  information obtained from, 134, 135–137
  monitor mode, 137
  OmniPeek, 135
  port scanning with, 105–106
  programs, 120–121
  requirements, 135
  Web sites, 356–358
  Wireshark, 135
Network Associates, 134
network browsing, UDP ports for, 188
Network File System (NFS), 220–221
network infrastructure attacks
  analyzers, 120–121
  ARP spoofing, 140–143
  banner grabbing, 130–131
  case study, 118
  defenses, 149–150
  denial of service, 145–147
  firewall rules, 131–134
  MAC address spoofing, 143–144
  network analyzers, 134–140
  overview, 15, 117
  port scanning, 122–128
  scanners, 120–121
  SNMP scanning, 128–130
  vulnerabilities, 119
  vulnerability assessment tools, 121
network interface card (NIC), 132
network mapping
  Google Groups, 51
  overview, 50
  privacy policies, 51–52
  Whois lookup, 50
Network Security Bible (Cole), 117
Network Security For Dummies (Cobb), 113, 257, 323, 326
Network Security Toolkit, 154, 355
Network users, 365
NFS (Network File System), 220–221
NFS attacks, 220–221
NGSSQuirrel, 307, 352, 364
NIC (network interface card), 132
Nigerian 419 e-mail fraud, 72
nipper, 133
NIST Guide to Enterprise Password Management, 359
NIST National Vulnerability Database, 55
NIST SP800-58 document, 363
NIST Special Publication 800-48, 351
Nmap. *See also* software and testing tools
  command-line options, 124
  Connect scan, 126
  FIN Stealth scan, 126
  Linux system testing with, 208, 214
  Null scan, 126
  ping sweeping with, 123–124
  port scanning with, 53, 120
  scanning Linux system with, 212
  SYN Stealth scan, 126
  system scanning with, 187
  UDP scan, 126
  Web site, 357
  Xmas Tree scan, 126
NmapWin, 55, 120, 357
NoLMHash registry key, 113
nontechnical attacks, 14–15

North American Electric Reliability Corporation (NERC), 13
Novell ConsoleOne utility, 243, 245
Novell NetMail, 256
Novell Netware
  admin account, renaming, 243
  auditing, 246
  bindery contexts, removing, 245–246
  cleartext packets, 242
  eDirectory browsing, disabling, 244–245
  intruder detection, 237–238
  overview, 229
  patching, 246
  port scanning, 231–233
  rconsole attacks, 233–236
  security risks, minimizing, 243–246
  security tools, 230
  server access methods, 231
  server-console access, 236
  servers, 230
  TCP/IP parameters, 246
  testing for rogue NLMs, 238–241
  testing tools, 356
  vulnerabilities, 229–230
Novell Patches and Security, 360
npasswd, 114
N-Stalker Web Application Security Scanner, 278, 293
N-Stealth Web Application Security Scanner, 364
NTAccess, 108, 359
null password, 101–102
null sessions
  configuration and user information, 192–194
  countermeasures, 194–195
  disabling, 114
  mapping, 191
  `net view` command, 192
  overview, 190

• O •

Objectif Securité, 87
OCTAVE methodology, 360
Oechslin, Philippe, 87
office layout and usage, physical security, 80–82
Official Internet Protocol Standards, 117
omnidirectional antenna, 155
OmniPeek
  finding hidden APs with, 166–167
  network analysis with, 106, 135
  port scanning with, 53
  viewing encrypted wireless traffic with, 164
  vulnerability assessment with, 121
  Web site, 357, 363, 365
  wireless network analysis with, 154
Online Hacker Jargon File, 354
online resources
  Bluetooth, 351
  certifications, 352
  database tools, 352
  exploit tools, 352
  general research tools, 353
  hacking, 354
  keyloggers, 354
  laws and regulations, 354
  Linux tools, 355
  live toolkits, 355
  log analysis, 355
  messaging-system testing tools, 355–356
  NetWare, 356
  network testing tools, 356–358
  password cracking tool, 358–359
  patch management, 359–360
  security education, 360
  security methods and models, 360
  source code analysis, 361
  storage testing tools, 361
  system hardening, 361–362
  user awareness and training, 362
  Voice over IP, 362–363
  vulnerability databases, 363
  Web applications, 363–364
  Windows, 364–365
  wireless networks, 365–366
open ports, scanning, 53–55
Open Source Security Testing Methodology Manual, 58, 360
OpenBSD, 15
`OPENROWSET` command, 305
OpenSSH, 210
operating system attacks, 15
operating system attacks, Linux
  buffer overflows, 223–224
  file permission attacks, 221–223

Download from Wow! eBook <www.wowebook.com>

operating system attacks, Linux *(continued)*
general security tests, 225–226
`hosts.equiv` file attacks, 218–220
multiplatform update managers, 227
NFS attacks, 220–221
overview, 207
patching, 226
physical security attacks, 224–225
`.rhosts` file attacks, 218–220
security tools, 208–209
system scanning, 209–212
vulnerabilities, 208
operating system attacks, Novell Netware
admin account, renaming, 243
auditing, 246
bindery contexts, removing, 245–246
cleartext packets, 242
eDirectory browsing, disabling, 244–245
intruder detection, 237–238
overview, 229
patching, 246
port scanning, 231–233
rconsole attacks, 233–236
security risks, minimizing, 243–246
security tools, 230
server access methods, 231
server-console access, 236
servers, 230
TCP/IP parameters, 246
testing for rogue NLMs, 238–241
testing tools, 356
vulnerabilities, 229–230
operating system attacks, Windows
authenticated scans, 205–206
missing patch exploitation, 198–205
NetBIOS, 187–190
null sessions, 190–195
overview, 181–182
scanning, 185–187
security tools, 182–184
share permissions, 196–198
testing tools, 364–365
vulnerabilities, 182
operating systems, securing, 113–114
ophcrack (password-cracking software), 92, 96, 99–101, 359
Ophcrack Live, 81
organizational password vulnerabilities, 86–88
Orinoco card, 154–155
Ounce Labs, 300, 361
outsourcing, 332–333, 350
OWASP WebGoat Project, 298, 360

• *p* •

packet signing, 242
Pandora, 92, 359
Pandora NetWare, 242, 356
Paros Proxy, 286, 364
`passfilt.dll`, 114
passwd+, 114
password cracking
blank, 101–102
brute-force attacks, 95–96
case study, 87
checking for null/blank passwords in NetWare, 101–102
countermeasures, 109–112
database hacking, 306–307
dictionary attacks, 94–95
inference, 90–91
keystroke logging, 103–104
with network analyzer, 105–106
password-protected files, 102–103
password-reset programs, 108–109
rainbow attacks, 96
rainbow cracking, 99
shoulder surfing, 85, 90
social engineering, 89–90
software, 92–94
tools, 358–359
Unix passwords with John the Ripper, 98
weak authentication, 91
weak BIOS passwords, 107
weak password storage, 104–105
Web sites, 358–359
Windows password with ophcrack, 99–101
Windows passwords with pwdump3 and John the Ripper, 96–98
Password Management Guideline document, 96
Password Safe, 359
password-protected files, 102–103
password-reset programs, 108–109

passwords. *See also* password cracking
divulging, 71
malicious users, 27
null, 101–102
overview, 85
policy considerations, 110–111
possible combinations, 99
storage locations by operating systems, 93–94
storing, 110
strong, 110–111
vulnerabilities, organizational, 86
vulnerabilities, technical, 88
weak storage, 104–105
patches, security
automating, 325–326
for Linux systems, 224–225
managing, 325
for password hacking, 112
tools, 325–326
Web sites, 359–360
Patent and Trademark Office, 353
Payment Card Industry Data Security Standard (PCI DSS), 13, 132, 354
pcAnywhere, 123
PDF documents, 48–49
PGP Whole Disk Encryption, 108, 362
Philippines, hacking ring in, 29
phishing. *See also* social engineering
dumpster diving, 67
overview, 66
phone systems, 68
in social engineering, 62
using the Internet, 67
phone systems, 68
PHRACK, 33, 354
physical security
case study, 77
exploiting weakness in, 27
factors in, 76
overview, 75
tailgating, 77
technical security and, 77
vulnerabilities, 75
physical security attacks
buildings, 78–79
Linux operating system, 224–225
network components and computers, 81–84
Novell Netware, 236
office layout and usage, 80–82
utilities, 79–80
wireless networks, 176–177
Ping of Death, 145
ping sweep, 123–124, 127–128
Point-to-Point Tunneling Protocol (PPTP), 164
POP3 (Post Office Protocol version 3), 123
Port 80 Software, 299, 364
port number listing, 357
port number lookup, 357
port scanners
in ethical hacking, 20
how it works, 124
NetScanTools Pro, 126–127
Nmap, 126
programs, 53
SuperScan, 125
port scanning
commonly hacked ports, 123
countermeasures, 127–128
information obtained from, 53–55, 124–125
Linux systems, 209–212
in network infrastructure attacks, 122
Novell Netware systems, 231–233
ping sweep, 123–124
tools, 53, 124–127
Windows systems, 185–186
ports, commonly hacked, 123
PortSentry, 213, 357
power failure, 79
power-protection equipment, 79
PPTP (Point-to-Point Tunneling Protocol), 164
pre-shared keys (PSKs), 162
Pretty Good Privacy (PGP), 22
Prism Test Utility, 175
privacy, respecting, 17
privacy policies, 51–52
Privacy Rights Clearinghouse, 338, 363
Proactive Password Auditor, 92, 95, 359
Proactive System Password Recovery, 92, 359
professional liability insurance, 35
Project RainbowCrack, 96
PromiscDetect, 106, 140, 357

promiscuous mode, 106, 134
pwdump3, 92, 96–98, 359
Pyn Logic, 362

• Q •

QualysGuard. *See also* software and testing tools
  database testing with, 304
  denial of service testing with, 146
  Linux system testing with, 209
  storage system testing with, 309
  vulnerability assessment with, 56–57, 121
  Web site, 352, 357, 365
  Windows system testing with, 184, 199
QualysGuard Suite, 56–57
Queensland DoS attack, 175–176
Quest Policy Authority, 269

• R •

rainbow cracking, 87, 96, 99
Rainbow tables, 359
RainbowCrack, 92, 359
RC4 encryption algorithm, 161
Rcon program, 356
Real-time Transport Protocol (RTP), 272
reCAPTCHA, 297
recycling bins, 81
Red Hat Enterprise Linux, 220, 227
Red Hat Linux Security Advisories, 360
Red Hat Package Manager, 227
red team, 37, 77
reformed hackers, 333
regedit, 143
related Google operator, 282
remote procedure calls, 123, 181
Remote tool, 230, 356
remote-administration software, 83
reports
  action items, 321–322
  methods, 320–322
  organizing information, 317–319
  prioritizing vulnerabilities in, 319–320
  securing, 320
residential phone, 68
reverse social engineering, 70
`.rhosts` file attacks, 218–220
rich Internet applications (RIAs), 298
RIPE Network Coordination Centre, 51, 353
risks, 18
rogue network, 27
rogue wireless devices
  AP characteristics of, 165–166
  countermeasures, 170
  detecting with WLAN analyzers, 165–168
  overview, 165
root directory, 218
RPC/DCE for Microsoft networks, 123
RPM Package Manager, 227
`.rtf` file, 311
RTP (Real-time Transport Protocol), 272

• S •

SANS, 113
scanners, 120–121
screens, locking, 84
script kiddies, 26, 28
ScriptLogic Patch Authority Ultimate, 326
search engines, 48, 67
SearchSecurity.com, 20
SeattleWireless Hardware Comparison page, 365
SEC filings, 67
SecureCRT, 261
SecureIIS, 300, 362
SecureWorks, 331
Securities and Exchange Commission, 353
SecurITree, 360
Security Accounts Manager (SAM) database, 93, 97
security assessment tools, 44
security auditing, 12
security awareness, 333–334
security by obscurity, 299
security infrastructure, assessing, 327–328
Security Innovation, 300
security measures
  account enumeration attacks, 257–260
  ARP poisoning, 144
  ARP spoofing, 144
  assessing security infrastructure, 327–328
  banner attacks, 257
  banner grabbing, 131

buffer overflows, 223–224
database attacks, 308–309
default configuration settings exploits, 177–178
default script attacks, 294
denial of service attacks, 146–147, 176
directory reversal attacks, 282–283
e-mail attachment attacks, 253
e-mail attacks, 266–267
e-mail bombs, 253–254
e-mail connection attacks, 253
e-mail header disclosures, 263
encrypted traffic attacks, 164–165
file permission attacks, 222–223
firewalls, 133–134
`hosts.equiv` file attacks, 219–220
implementing, 323–324
input filtering attacks, 291
instant messaging vulnerabilities, 268–269
MAC address spoofing, 144, 175
missing patch exploitation, 205
NetBIOS, 190
Netware intruders, 238
NetWare Loadable Module, 241
network analyzers, 139–140
network infrastructure attacks, 135
NFS attacks, 221
null sessions, 194–195
packet capture, 242
password cracking, 109–114
patching, 324–326
physical security attacks, 224–225
physical security problems, 176–177
port scanning, 127–128
`.rhost` file attacks, 219–220
rogue NLM attack, 241
rogue wireless devices, 170
security awareness and training, 333–334
SMTP relay attacks, 263
SNMP scanning, 130
social engineering, 72–74
storage system attacks, 313
system hardening, 326–327
system scans, 212
unneeded services, 216–217
unsecured login mechanisms, 297
Voice over IP, 276
vulnerable wireless workstations, 177–178
wireless network attacks, 158–159
Security On Wheels, 360
security portals, 20
Security Tools Distribution, 154, 355
SecurityFocus.com, 20
semidirectional antenna, 155
sendmail server, 214
Server Message Block (SMB), 188
ServerDefender, 300, 362
ServerMask, 299, 364
service set identifier (SSID), 157–158
Session Initiation Protocol (SIP), 272
SetGID, 221–222
setpwd password reset tool (NetWare Loadable Module), 238
SetUID, 221–222
share permissions
  defaults, 196
  overview, 196–198
  testing, 197
  Windows 2000/NT, 196
  Windows XP, 196
ShareEnum, 184
Shavlik Technologoes NetChk, 325
shoulder surfing, 85, 90
shredders, 67
Sima, Caleb (SPI Dynamics), 279
Simple Mail Transfer Protocol (SMTP), 16, 123, 257
Simple Network Management Protocol (SNMP), 123, 128–130
SIP (Session Initiation Protocol), 272
sipsak, 363
SiteDigger, 364
site:hostname keywords: query (Google), 281
SiVus, 272–274, 363
Slackware, 154, 227
Slackware Package Tool, 227
SMAC, 143–144, 173, 357
SMB (Server Message Block), 188
SMTP (Simple Mail Transfer Protocol), 123, 257
SMTP attacks
  account enumeration, 257–260
  e-mail header disclosures, 263–264
  e-mail traffic capture, 264
  malware, 264–266
  relay, 260–263

smtpscan, 256, 356
Smurf, 160, 351
SNARE, 357
Sniffdet, 140, 358
sniffers, 134
SNMP (Simple Network Management Protocol), 123, 128–130
SNMP scanning, 128–130
SNMPUTIL, 128, 358
social engineering
  behaviors associated with, 69–70
  believability in, 69
  building trust in, 68–69
  case study, 63
  consequences of, 65
  countermeasures, 72–74
  deceptive practices in, 69–72
  defined, 15
  examples of, 61–62
  false employees, 62
  false support personnel, 62
  false vendors, 62
  likability in, 69
  outsourcing, 64
  overview, 61
  password cracking, 89–90
  phishing, 62, 66–68
  policies, 72
  reasons for using, 64–65
  reverse, 70
  user awareness and training, 73–74
software and testing tools
  Linux systems, 208–209
  network analyzers, 120–121, 135
  Novell Netware, 230
  password cracking, 92–94
  scanners, 120–121
  selecting, 20–21, 44
  storage systems, 309–310
  vulnerability assessment, 121
  Web applications, 278
  Windows systems, 181–185
  WLAN security tools, 154–155
Software as a Service (SaaS), 331
Software Engineering Institute, 360
SonicWall, 300, 313
source code analysis, 300–301, 361
SourceForge.net, 209
Southeast Cybercrime Institute, 251
Special Ops Security, 306
Spector Pro (keystroke-logging software), 104
SpectorSoft, 104, 354
spidering, 278, 280–281
Spitzner, Lance, 34
sponsorship, 18
SQL injection, 27, 287–289
SQL Injector, 288–289
SQLPing3 (password-cracking software), 93, 304, 352, 359
SSH (Secure Shell), 123
SSID (service set identifier), 157–158
storage system attacks. *See also* database attacks
  best practices for minimizing risks, 313
  misconceptions, 309
  overview, 309
  scanning for vulnerabilities, 310
  testing tools, 309–310, 361
  text file search, 310–313
  Web sites, 361
StorScan, 310, 361
strong passwords, 110–111
SUN RPC (remote procedure calls), 123
Super Cantenna, 365
SuperScan
  Linux system testing with, 208, 209–212
  Netware testing with, 230
  pinging multiple addresses with, 52
  port scanning with, 53, 120
  scanning Novell Netware systems with, 232
  storage system testing with, 309
  Web site, 358, 361
  Windows system testing with, 184
SUSE Linux, 227
SUSE Linux Security Alerts, 360
.swf files, 48–49
SWFScan, 298, 364
Switchboard.com, 353
SYN floods, 145
Sysinternals, 183, 365
SYSKEY utility, 113
System Center ConfigurationManager, 269
system hardening, 326–327, 361–362
system scans. *See also* port scanning

countermeasures, 212
hosts, 52
information obtained from, 52–53
Linux systems, 209–212
scanning, 53–55
Windows systems, 185–187
systems
knowledge of, 19
selecting, 18

• T •

tailgating, 77
TCP ports, 188
TCP scans, 122
TCP Wrappers, 217, 358
tcpcon (NetWare Loadable Module), 239–240
TCP/IP For Dummies (Leiden), 117
TCPView, 184
telecom wires, 80
Telnet, 123, 130–131, 209, 213
Temporal Key Integrity Protocol (TKIP) encryption, 162
tests
assumptions in, 43–44
blind vs. knowledge assessments, 42
denial of service, 146
Linux systems, 225–226
location, 43
MAC address protocols, 171–174
mistakes in, 347–350
Netware intruders, 237–238
Novell Netware intruders, 237–238
overview, 40
password security, 97–100
performing, 45–47
reacting to vulnerabilities, 43
rogue NLM programs, 238–241
share permissions, 197
specific tests, 41–42
standards, 40
timing, 40–41
VoIP hosts, 273
Windows systems, 185–187
text files, searching for, 310–313
TFTP (Trivial File Transfer Protocol), 123
THC-Amap, 208
TheHarvester, 258
TheTrainingCo, 77
Tiger, 208
tiger team, 37
time-memory tradeoffs, 87
Traffic IQ Pro, 132–133, 358
Tripwire, 223
Trivial File Transfer Protocol (TFTP), 123
TrueCrypt, 108, 224, 362
trustworthiness, 16
TSGrinder, 359
Twitter, 145
.txt file, 311

• U •

UAC (User Account Control), 198
UDP ports, 188
UDP scans, 122
UDPFlood, 146, 358
unauthorized software, 27
Universal Naming Convention (UNC), 306
Unix systems
password protection in, 114
wireless hacking tools for, 154
unlimited attack, 19
unneeded services
access control, 217
`chkconfig`, 217
countermeasures, 216–217
disabling, 216–217
`inetd.conf`, 216–217
security tools, 214–216
vulnerabilities, 213–214
unsecured login mechanisms, 294–297
up2date, 227
URL manipulation, 285
U.S. Patent and Trademark Office, 353
US Search.com, 353
US-CERT Vulnerability Notes Database, 55, 363

User Account Control (UAC), 198
user awareness and training, 362
user ID, 27, 86, 112
UserDump, 356
USSearch, 49
utilities, physical security, 79–80

• V •

VBScript, 290
vendor passwords, default, 358
virtual local area network (VLAN), 271
virtual machine software, 52
virtual private network (VPN), 164
VirtualBox, 52, 148
VMWare Workstation, 52
VNC, 83
Voice over IP (VoIP). *See also* messaging-system attacks
  attacks, 16
  capturing and recording voice traffic, 274–276
  countermeasures against vulnerabilities, 276
  overview, 16, 270
  scanning for vulnerabilities, 272–274
  testing tools, 362–363
  vulnerabilities, 270–272
  Web sites, 362–363
VoIP For Dummies (Kelly), 270
VoIP Hopper, 271, 363
VoIP servers, 68
vomit, 363
`VRFY` command, 257, 259
vulnerabilities. *See also* vulnerability testing
  addressing, 323
  assessing, 55–57
  database attacks, 307–308
  high-impact, 320
  instant messaging, 267–268
  Linux systems, 208
  low-impact, 320
  medium-impact, 320
  messaging-system, 249–250
  network infrastructure, 119–120
  Novell Netware, 229–230
  passwords, 86–88
  physical security, 76
  ranking, 320
  reporting, 319–320
  storage systems, 310
  unneeded services, 213–214
  Voice over IP, 270–272
  Web applications, 280, 297
  Windows systems, 182
  wireless local area networks, 152
vulnerability assessment tools, 121
vulnerability databases, 363
vulnerability scanners, 17, 146
vulnerability testing
  automated assessment, 56
  manual assessment, 56
  tools, 56–57
  Web sites, 55

• W •

walls, 79–80
wardriving, 155
warwalking, 168
weak authentication, 91
Web 2.0 hacking, 297
Web application attacks
  best practices for minimizing risks, 298–301
  case study, 279
  default script attacks, 292–293
  directory reversal, 280–283
  input filtering attacks, 283–291
  scanning for vulnerabilities, 297
  testing tools, 20, 278
  tools, 363–364
  unsecured login mechanisms, 294–297
  vulnerabilities, 280
  Web sites, 363–364
Web browsers, configuring for Web proxy, 284
Web crawling, 49
Web page defacement, 31
Web Proxy, 284, 286

Web search, 48
Web security
  firewalls, 299–300
  obscurity, 299
  source code analysis, 300–301
Web sites
  background checks, 49
  Bing, 48
  Bluetooth, 351
  certifications, 352
  database tools, 352
  defaced Web pages, 31
  exploit tools, 352
  general research tools, 353
  Google, 48
  government and business, 49
  hacking, 354
  keyloggers, 354
  laws and regulations, 354
  Linux tools, 355
  live toolkits, 355
  log analysis, 355
  messaging-system testing tools, 355–356
  NetWare, 356
  network analyzers, 121–122
  network testing tools, 356–358
  password cracking tools, 358–359
  patch management, 359–360
  scanners, 121–122
  security education, 360
  security methods and models, 360
  security portals, 20
  security tools, 20
  source code analysis, 361
  Spitzner, Lance, 34
  storage testing tools, 361
  system hardening, 361–362
  user awareness and training, 362
  VirtualBox, 52
  Voice over IP, 362–363
  vulnerability assessment tools, 122
  vulnerability databases, 363
  vulnerability testing, 55
  Web applications, 363–364
  Whois lookup, 50–51
  Windows, 364–365
  wireless networks, 365–366
WebGoat, 364
WebInspect, 146, 278, 288, 290, 364
Wellenreiter, 154, 366
WEP (Wired Equivalent Privacy), 160–161
WEPCrack, 161, 366
WhatIsMyIp.com, 52, 353, 358
white hat hackers, 10
Whois.net, 50, 353
Whole Disk Encryption, 84, 108
Wi-Fi. *See* wireless local area networks (WLANs)
Wi-Fi Protected Access (WPA), 160, 162
WifiMaps, 366
WiGLE database, 156–157, 366
WildPackets OmniPeek, 121, 154
Wiles, Jack (TheTrainingCo.), 77
WinAirsnort, 366
windows, 79–80
Windows 7 operating system, 198
Windows BitLocker, 84, 108
Windows Defender, 198
Windows Firewall, 198
Windows operating system
  attacks, 15
  password storage location in, 93–94
  securing, 113–114
Windows password. *See also* password cracking
  cracking, 87
  cracking with ophcrack, 99–101
  cracking with pwdump and John the Ripper, 96–98
  protection of, 113–114
Windows Registry, 113, 143–144
Windows Server Update Services, 326, 360
Windows systems. *See also* Linux systems
  authenticated scans, 205–206
  missing patch exploitation, 198–205
  NetBIOS, 187–190
  null sessions, 190–195
  overview, 181–182
  scanning, 185–187
  security tools, 182–184
  share permissions, 196–198
  testing tools, 364–365
  vulnerabilities, 182
Windows Terminal Server, 123

Winfo, 184, 192–193, 365
WinHex, 111, 292, 359
Winkler, Ira (Internet Security Advisors Group), 63
WinNuke, 145
WinRAR, 95
WinZip, 97, 356
Wired Equivalent Privacy (WEP), 160–161
wireless antennas, 155
Wireless Hardware Comparison, 155
wireless local area networks (WLANs)
  access points, 152
  case study, 153
  checking for AP's MAC address, 156–157
  default configuration settings, 178
  hacking tools, 154–155
  overview, 151
  scanning SSIDs, 157–158
  vulnerabilities, 152
  WiGLE database, 156–157
wireless network attacks
  Bluetooth, 160
  encrypted traffic, 160–165
  MAC address spoofing, 170–175
  overview, 158–159
  physical security problems, 176–177
  Queensland DoS attack, 175–176
  rogue wireless devices, 165–170
  tools, 365–366
  vulnerable wireless workstations, 177–178
  Web sites, 365–366
Wireless Vulnerabilities and Exploits, 152, 363
wireless workstations, vulnerability of, 177–178
Wireshark, 53, 106, 135, 268, 358
word lists (password cracking), 358
Wotsit's Fromat, 353
WPA (Wi-Fi Protected Acess), 160, 162
WPA2, 162–163
Wright, Joshua (InGuardians Inc.), 153
WSDigger, 298, 364
WSFuzzer, 298, 364

• X •

`.xls` file, 311
`.xlsx` file, 311
`xp_dirtree` extended stored procedure, 306
XSS (cross-site scripting), 290

• Y •

Yahoo! Finance, 353
YaST2 Package Manager, 227

• Z •

ZabaSearch, 49, 353
zombie computers, 71

Get out your white hat and learn where your systems may be vulnerable

You're a good guy or gal, so why do you need to learn how to hack? Because the only way to be sure your systems are secure is to find out how the bad guys work and examine your defenses from their point of view. This guide shows you how, explains common attacks, tells you what to look for, and gives you the tools to safeguard your sensitive business information.

- ***Build the foundation** — understand the value of ethical hacking, what's involved, and the malicious hacker's mindset*
- ***Games people play** — discover how hackers use social engineering to breach security and what to do about it*
- ***It's the network** — explore common network vulnerabilities and the creative ways they're exploited*
- ***Down and dirty OS hacking** — learn how Windows, Linux, and Novell NetWare are being attacked and how to scan for vulnerabilities*
- ***Sneak attacks** — see why applications, especially Web apps, are vulnerable and how to protect them*
- ***Get the message** — prepare for attacks on e-mail, IM, and VoIP systems*
- ***Tools of the trade** — learn about Metasploit, BackTrack, and other important security testing tools*
- ***Now what?** — find out how to use the information you gather to minimize business risks*

Open the book and find:

- What makes a hacker hack
- Why you need to hack your systems
- How to gain management's approval for your ethical hacking tests
- Countermeasures to common attacks
- Linux and Novell NetWare risks
- Techniques for defending databases
- How wireless LANs are compromised
- Ten deadly mistakes to avoid

Go to Dummies.com®
for videos, step-by-step examples, how-to articles, or to shop!

Kevin Beaver is an independent information security consultant, expert witness, and speaker with more than 20 years of security experience. He specializes in performing information security assessments that support compliance and risk management. He is also coauthor of *Hacking Wireless Networks For Dummies*.

For Dummies®
A Branded Imprint of
WILEY

$29.99 US / $35.99 CN / £21.99 UK

ISBN 978-0-470-55093-9

52999

9 780470 550939